



PUBLIC
2024-04-16

SAP Mobile Services (Cloud Foundry) Preview

Content

1	SAP Mobile Services (Cloud Foundry)	3
1.1	Feature Scope (Cloud Foundry)	3
1.2	What Is Mobile Services?	14
1.3	What's New (Cloud Foundry)	15
1.4	Get Started	18
	Enable Developer Accounts	19
	Set Up Customer Accounts	20
	Using Mobile Services Cockpit in an Extended Landscape	29
1.5	Administration	30
	SAP Mobile Services Overview	31
	User Interface	39
	Getting Support From the Cockpit	42
	Accessibility	43
	Application Administration	44
	Security	281
	Troubleshooting: Common Issues	318
1.6	Development	323
	REST API Application Development Overview	323
1.7	Migration	529
	Migrate: SAP Mobile Services Neo to Cloud Foundry	530
	Migrate: SAP Mobile Platform to SAP Mobile Services	533
	Upgrading Apps to Mobile Services Cloud Foundry Service	535
	New Feature Comparison: Cloud Foundry and Neo	536
1.8	Glossary	537

1 SAP Mobile Services (Cloud Foundry)

SAP Mobile Services is an open, standards-based cloud platform that enables simple mobile application development, configuration, and management.

You can preview new Mobile Services features for a week or so before they are released for production. This gives you an opportunity to test new features in advance. Note that the preview landscape is not intended as a development or production environment.

Mobile Services Preview is only available on the eu10 landscape (AWS Frankfurt). It is available as its own service offering from the Service Marketplace. When using preview, a banner in Mobile Services cockpit indicates "Preview" and a notification provides a reminder that Preview is not for production or development.

See: [SAP Mobile Services \(Cloud Foundry\) Preview](#), *What's New (Cloud Foundry)* (in PDF format)

📘 Note

You can access the Preview documentation from the SAP Mobile Services Product Page.

1.1 Feature Scope (Cloud Foundry)

Summarizes the core features included for SAP Mobile Services running in a Cloud Foundry landscape.

Mobile Services supports development for multiple app types, namely: native apps, hybrid apps, and web apps. Define, manage, and monitor your apps through their entire life cycles.

App Catalog

Manage your apps during development, which allows you to easily test-drive apps with test users. Use over-the-air (OTA) distribution to bring new releases to testers, and when you're done with quality assurance, push releases to your enterprise mobility management (EMM) solution to bring them to your end users.

App Catalog Features

Feature	Description
Manage app versions	Manage Android and iOS apps and versions from various sources.
OTA downloads	Generate download links and QR-codes that can be sent to testers.
Cloud Build integration	Publish custom clients and apps built with Cloud Build.
External build tool integration	Use Service Keys to integrate external build tooling.

Feature	Description
Upload apps	Manually upload apps from other sources.
Push to EMM	Upload apps from your app catalog to EMM for end user distribution.

Application Update

Manage each application through all phases of its life cycle.

Application Update Features

Feature	Description
Kapsel-based apps support	Manage and roll out updates for Kapsel-based applications.
Mobile Development Kit-based apps support	Manage and roll out updates for apps built with Mobile Development Kit.
Staging support	Developers can upload new versions but restrict distribution to a predefined list of test users, before publishing to production.
Lifecycle events	Various events are exposed to the app developer to control the behavior of the app. This allows a developer to let the user delay an update or force an update of the app, for example.
Manual upload of app versions	Developers can manually upload new versions of their app using Mobile Services cockpit and APIs.
Command-line support	Various command-line tools help to automate app lifecycle management.
Exposed RESTful API	RESTful API for custom automation of app lifecycle management.

Application Links

Seamlessly deep link from web pages to app content, which eases the transition from easily accessible web content to truly native experiences.

Application Links Features

Feature	Description
Configure Universal Links	Enable Universal Links for your iOS apps by providing your team and bundle ID.
Configure app links	Enable app links for your Android apps by providing your association file.

Back-end Connectivity

Enable mobile apps to connect securely with data and systems through back-end connectivity, and to manage service versions, control system load, and protect data.

Back-end Connectivity Features

Feature	Description
Manage destinations	Securely connect your apps to business data via HTTP(S).
Authentication	Authenticate back-end access by means of Basic (including SCIM), App-to-App SSO, Forward Authentication, Cloud Connector SSO, OAuth2 SAML Bearer Assertion or Client Certificates.
Virus scanning	Scan incoming and outgoing traffic for harmful code and data.
URL rewriting	Automatically rewrite URLs, such as OData entity URIs or relative URLs used by apps.
Control system load	Define limits for request size, concurrent connections, time-outs and requests-per-second that your back end should handle.
User name propagation	Additionally propagate the user name to back ends by means of headers.
Add custom headers to requests	Add additional headers to requests through Mobile Services, such as SAP Business Accelerator Hub API keys.
Optimized for OData	Auto-generate code in SAP BTP SDK for Android and iOS.
Add annotations to services	Add additional metadata to your OData, for example, to enable tools to automatically generate UIs.
Test destinations	Check service availability and browse OData back ends from the Mobile Services cockpit.

Client Resources

Enables mobile apps to offer on-device access to important business systems, and to attachments and other binaries across multiple mobile devices. Use this feature to provide your users with static content, machine learning models trained in the cloud, and themes for apps built with the Mobile Development Kit.

Client Resources Features

Feature	Description
Manage bundles	Upload and version, download or view files and file archives in the Mobile Services cockpit.

Feature	Description
Distribute bundles	Download and apply bundles and new bundle versions to your apps using our SDKs.

Cloud Build

Provides the means to easily build clients using the latest SAP SDKs in the cloud, without having to install any developer tooling. This way, everyone can easily tweak app names and assets without the associated cost of maintaining code bases.

Cloud Build Features

Feature	Description
Build custom clients	Generate and build up-to-date Asset Manager, Mobile Development Kit Client and Customized Mobile Development Kit Clients with custom branding and settings.
Over-the-air app distribution	Use and share generated links and QR codes to install built apps on test devices.
Manage signing profiles	Generate or upload Android and iOS signing profiles for distribution signing.
Purge old artifacts	Automatically remove old build logs and binaries that are no longer required.
Package hybrid apps	Package UI5 app assets from SAP Web IDE into Cordova containers for offline enablement.

Discovery Service

Enables you to use what users know or can access more easily, such as their email domain or an onboarding code, to connect to the right system landscape when first using the app, without requiring custom code lines per environment.

Discovery Service Features

Feature	Description
Claim domains for onboarding	Use the Mobile Services cockpit to claim your domain for onboarding.
Generate onboarding codes	Generate onboarding codes for your users.
Publish app configurations	Enable individual applications for runtime discovery.
Configure custom discovery	Customize configuration returned to apps by Discovery Service.

Feature Restrictions

Use Feature Restriction policies to ship early features without having to immediately enable them, to enable specific sets of features per landscape or customer, or simply to turn off certain features in apps using SAP SDKs.

Feature Restrictions Features

Feature	Description
Restrict features in Kapsel-based apps	Disable camera, bar code scanner, contacts, file, location, calendar, printer or push in your apps.
Manage feature flags	Create custom feature flags and use them to control arbitrary features in your apps.

Fiori

Equips designers and developers with a set of tools and guidelines to create apps for any platform, providing a consistent, innovative experience for both creators and users. The SAP Fiori for Android and iOS design languages take the strengths of the Fiori user interface and the mobile operating systems to quickly deliver enterprise applications. Additionally, apps built with the Mobile Development Kit come with native Fiori controls by default, without any additional effort required.

Fiori Features

Feature	Description
Design consistent user experiences	Use stencils to sketch Fiori user interfaces.
Explore and configure Fiori controls	Use the Fiori Mentor app to tweak Fiori control settings, side-by-side with the code your app needs.
Implement native Fiori for your app	Use the SAP BTP SDK for Android or iOS to add native Fiori controls to your apps.
Out-of-the-box native Fiori with Mobile Development Kit	Apps built with the Mobile Development kit use native Fiori controls by default.

Logging

Enables you to gain insights into live apps that are rolled out to a large number of users. Use our SDKs to gather logs on client devices, to add contextual information and to automatically or manually upload them to Mobile Services for further inspection. Leverage the Mobile Services cockpit to gain an integrated insight into your user base, regardless of the platform on which the issue originally occurred, and to access raw log data for further analysis in dedicated tools.

Logging Features

Feature	Description
Gather client logs	The SAP BTP SDK and Mobile Development Kit by default log certain actions, which can be made visible by means of configuration. In addition, they allow you to log custom messages on different levels of severity. Both standard SDK and custom logs can be uploaded to Mobile Services for remote inspection and analysis.
Gather server logs	Mobile Services has a wide range of logging options that can be configured on the component level. Use the web interface to set log levels per component.
View technical and event logs	Both uploaded client and server-side logs can be inspected in a dedicated web interface that offers an array of filters to bring order to the variety of logs gathered. In addition, event logs are provided, which aggregate complex business operations that would otherwise result in an abundance of separate service calls and hence technical log entries.
Download gathered logs	For further analysis with specialized tools, Mobile Services lets you download arbitrary subsets of gathered logs to your computer.
Archive logs	In order to focus on recent events, Mobile Services automatically archives logs that exceed a certain threshold. Use the dedicated log settings to configure what kind of logs should be archived, and when.
Auditing	Traceability of changes in an enterprise world is key, which is why Mobile Services by default tracks changes to all settings and exposes an activity log that can be filtered in a number of ways. An export option is available for further processing.

Offline

Design applications to run in offline-mode, enabling users to download business information and data periodically when network connections are available, and make changes when connections are not available. When offline, switch to the local data source, which is kept current via OData synchronization features and open standards.

Offline Features

Feature	Description
CRUD	OData offline supports full create, read, update and delete (CRUD) operations while the user is offline.
View offline configuration settings	View offline configuration settings in the Mobile Services cockpit.

Feature	Description
View usage statistics for offline OData applications	Administrators can view request and response-time usage statistics for offline OData applications in the Mobile Services cockpit. Statistics are gathered for offline data store operations such as build, refresh (download) and flush (upload).
Offline data security	The local data storage used for offline access is encrypted on the device. When the Offline Store Upload API is implemented, users can securely upload local database files to the server.
OData v2 data sources	Offline OData supports access to OData services following the v2 specifications, with additional v4 metadata annotations.
OData v4 lambda operator	Offline store supports OData v4 lambda operators any and all.
Conflict detection	The use of ETags allows conflict detection, enabling developers to notice data modifications on the same entity and react accordingly in their app.
Handling of failed requests	Allows developers to write robust apps that can recover from business logic errors that happened while offline.
Media resources	OData Offline supports handling of media resources provided by the back-end service.
Repeatable requests	To ensure data consistency, we support the prevention of repeatable requests in Offline OData.
Upload of local store	For root cause analysis, developers can extend their apps to upload the local data store to Mobile Services.
CLI tools	Local tooling for troubleshooting offline scenarios.
Event Log	Local-only entity set to view the log of past system offline system events.
Progress API	Enables developers to inform users about ongoing data synchronization.
Request Queue Optimization	Built-in heuristics that reduce the number of requests that get sent to the back end.
Transaction Builder	Allows grouping of CUD operations into transactions (OData change sets).

Feature	Description
Undo changes	Undo local modifications of entities before uploading to the back end.
Complex objects graphs	Allows the local creation of complex entity relations while being offline.

Onboarding

Provides a smooth user onboarding process that enables users to become effective using your software without any external effort. Mobile Services and its associated tools provide several means to achieve efficient onboarding, which includes making it easier for users to find your apps, to help them connect to the right systems, and to understand how to use them.

Onboarding Features

Feature	Description
Fiori everywhere	Make it easier for users to understand your app by implementing common Fiori patterns.
Consistent onboarding experience	Use Fiori Flows for a native Fiori onboarding experience across apps.
Manage terms and consent	Use Fiori Flows to communicate usage terms and to ask for user consent.
Auto-configure apps	Use Discovery Service to enable apps to auto-configure themselves.
Seamless transition between web and mobile	Use application links to guide users from web pages to native experiences.

Push

Proactively notify your mobile users of important events using a variety of scenarios. The primary advantage of the Mobile Services implementation of push is that mobile solution developers do not need to implement specific code for APNs (Apple Push Notification Services) or FCM (Firebase Cloud Messaging). Instead, Mobile Services exposes a consistent API to the event source (back end). Mobile Services also provides predefined push configurations to allow SAP-delivered applications, available via public app stores, to deliver notifications. The back-end-facing API of the push feature offers different ways to reach segments of your user base and abstract from native push providers, and exposes various platform-specific features as well.

Push Features

Feature	Description
Push Notification	For mobile applications, the platform manages the certificates, tokens, and push notifications for individual applications. When changes occur, the back end can send push notifications to mobile applications on devices that are push-enabled.
Push Desk Notification	Administrators can send push notifications to all users of a push-enabled application from the Mobile Services cockpit. You can check all registered users and devices and select an individual user to whom to send a push message. This functionality is mostly used to test the end-to-end setup and configuration of the push functionality.
Predefined Push Configuration	You can enable or disable preconfigured push settings for SAP-delivered applications. When enabled, the default push configuration that comes with the Apple App Store and Google Play version of the app is used. When disabled, you can configure push settings manually.
Push Statistics	Provides statistics about push notifications being sent out to mobile apps.
Capabilities-based Push Support	This feature adds an abstraction layer to identify different recipient apps on the user's device. It basically holds information about which app to notify about a certain event.

Security

Enables you to provides secure propagation of mobile users' identities to back-end systems. Supports a range of popular application authentication protocols and maps them to back-end systems. Additionally, Mobile Services and its related tools provide a variety of means to secure data at rest and in motion.

Security Features

Feature	Description
Authenticate users	Authenticate using popular protocols such as OAuth, SAML or Basic authentication.
Propagate user identities	Forward user authentication by means of Basic, Application-to-Application SSO, Forward Authentication, Cloud Connector SSO or OAuth2 SAML Bearer Assertion.
Virus Scanning	Scan inbound and outbound traffic for harmful code and data.
Role-based access	Secure your applications by creating roles and assigning them to users.

Feature	Description
User blocking	Prevent specific users from further using existing apps or from accessing new ones.
User locking	Prevent users from using apps that haven't connected to Mobile Services for a certain period of time.
Data wiping	Request deletion of client-side data of clients that haven't connected for a certain period of time.
Automatic user removal	Force re-registration of clients that haven't connected for a certain period of time.
Cross-site request forgery (CSRF) protection	Protect users from CSRF attacks.
Encrypted client data	Protect data in apps by means of passcodes and biometrics.
Multi-landscape support	Build pure cloud solutions or connect to on-premise back ends in hybrid cloud scenarios.

Storage Service

A flexible, scoped key-value store that is used in a number of Mobile Services features, and that can be used in applications to store various developer-defined information.

- Application-level storage can be used to apply shared configuration to all app installations, such as general policies.
- The finer-grained user-level storage can hold user preferences, draft objects and other data that should be available across devices.
- Device-level storage can be used for information pertaining to the device or installation.

The application configuration data is stored based on user or device preference. The storage service stores flexible data structure and supports application-level, user-level, and device-level storage. Mobile Services offers authorization and authentication schemes that secure the data..

Storage Service Features

Feature	Description
Key-value store	Flexibly store various information in the cloud.
Scoped access	Use application-wide scope to configure all installations, or use user-level and device-level storage for more specific information.

Tracing

The Network Trace feature allows developers, supporters and administrator to inspect the network traffic flowing between apps and Mobile Services, and between Mobile Services and other connected services. It is a powerful tool to localize issues, and includes an export function to easily replay requests in isolation.

Tracing Features

Feature	Description
Capture HTTP traffic	Make recordings of incoming and outgoing requests between Mobile Services, apps and other services.
Fine-grained recordings	Limit recordings to specific users, content types, targets (apps or other services).
Capture different scopes	Choose to capture only request headers or both headers and message body.
HTTP Archive (HAR) exports	Download recordings as HAR files to view them locally or in dedicated tooling.

Translation Hub

Streamline translation routines and tap into new markets by localizing products. Translate texts using APIs or integrated translation workflow scenarios that access a multilingual database and machine translation capabilities. Meet the demands of your industry- or company-specific terminology needs by uploading and using your own language data, whether you are using HTML-based apps, mobile apps or traditional ABAP user interfaces. You can find more information on Translation Hub on the [Capabilities](#) site.

Translation Hub Features

Feature	Description
Connect mobile projects to SAP Translation Hub	Use existing SAP Translation Hub deployments and projects to translate project resources.
Integrated with local tooling	Use the SAP BTP SDK for Android Wizard or the iOS Assistant to access Translation Hub from your workstation.

Usage

The Mobile Services Client Usage and User Feedback service, powered by SAP Analytics Cloud, provides insights into user behavior which are required to evolve your apps in a meaningful way. On top of the collected usage data from your apps Mobile Services provides you with a set of prepared reports which helps you to identify improvement areas of your apps as well as measuring the success of your apps. These insights allow you to streamline development resources to areas of improvement and provide data-driven decisions for further investment or disinvestment.

Usage Features

Feature	Description
Collect usage data	Collect standard and custom data on user behavior through SAP Mobile Cards, SAP Mobile Development Kit, and the SAP BTP SDK for Android and iOS.
Gather user feedback	Receive user ratings and comments through SAP Mobile Development Kit and the SAP BTP SDK for Android and iOS.

Feature	Description
Server data reports	Analyze server performance data to optimize app operations.
Integrated with SAP Analytics Cloud	Analyze usage and feedback data in powerful reports built on top of SAP Analytics Cloud.
Network policy	Restrict upload based on network conditions to minimize impact on cellular data.

User Information

Allows developers to query Mobile Services for information about the currently logged-in user in order to display personalized data. Additionally, when used in conjunction with Access Control policies, developers can access user role information in order to deliver role-based experiences and control access to app features. If a user does not have the required role and tries to register for an application, the access control policy returns a 403 error message.

User Information Features

Feature	Description
Read user information	Access information such as the actual user name and email address through the Cross-domain Identity Management (SCIM) protocol.
Build role-based apps	Leverage roles defined for users to build role-based screens.
Multiple authentication schemas	Expose user information in apps using SAML or OAuth authentication.

1.2 What Is Mobile Services?

Learn more about SAP Mobile Services for SAP Business Technology Platform (SAP BTP). Enables simple mobile application development, configuration, and management.

SAP Mobile Services enable you to develop, configure, and manage mobile applications that provide mobile access to enterprise data. Mobile Services key features include: app content lifecycle management, push notifications for timely enterprise data updates, support for on-device storage of enterprise data (offline apps), app security, and app monitoring and usage reporting.

SAP Mobile Services is a cloud-based offering that is complemented by various development tools such as the native SDKs for iOS, Android, the mobile development kit or SAP Mobile Cards. The overall objective is to deliver reusable services, which can be leveraged in mobile app development projects. SAP BTP provides a robust foundation on which to base an enterprise-wide mobile strategy for both in-house and consumer-facing mobile solutions.

Features

Mobile Services	Supports development for multiple app types, namely: native apps, cross-platform apps, web apps, and Mobile Development Kit clients. Define, manage, and monitor your apps through their entire life cycles.
App Updates	Intelligent update mechanisms keep your apps and app configurations up-to-date through optimized downloads, and allow you to maintain multiple app versions in the field.
Work Offline	When offline, switch to the local data source, which is kept current via OData synchronization features and open standards.
Security	Choose the authentication types that suit your data protection needs from multiple options.
Push	Send updates and notifications from the back-end data source to mobile apps. Use the push desk to send ad-hoc notifications to users of a specific app.
Usage Reporting	View app-specific usage analytics, statistics, and reports that comply with GDPR standards.
Build or Extend	Extend existing on-premise solutions, or use SAP BTP to build a fully cloud-hosted solution that uses HANA as the back end, the Java Runtime, HTML5 app, and mobile app tools.

SAP Mobile Services is a cloud-based offering that is complemented by various development tools such as the native SDKs for iOS, Android, the mobile development kit or SAP Mobile Cards. The overall objective is to deliver reusable services, which can be leveraged in mobile app development projects. SAP BTP provides a robust foundation on which to base an enterprise-wide mobile strategy for both in-house and consumer-facing mobile solutions.


Environment

This service is available in the Cloud Foundry environment.

1.3 What's New (Cloud Foundry)

New features and feature enhancements in SAP Mobile Services.

See also:

- [What's New for SAP Business Technology Platform](#) for an overview of new and changed features for SAP BTP. Use filters to find specific information.
- For information about support packages, patches, and fixes (log on required):
 - <https://launchpad.support.sap.com/#/notes/3213447>  (Preview)

- [SAP Note 3096385](#)  (Production)

New Features - April 2024 (Version: 2404)

Features	Type of Change	Description
OData destination quality check	New	<p>You can now test the quality of an OData destination connection. The test provides information that can help you improve the back-end connection during development. The Offline feature must be enabled for the application.</p> <p>See:</p> <ul style="list-style-type: none"> • <i>Creating a Destination</i> • <i>Testing OData Destination Quality</i> • <i>Editing the Application Configuration File</i>
Certificate expiration alerts	New	<p>For Mobile Services that require a certificate, you can now define a certificate expiration alert in the Alert Notification Service (ANS). This enables you to control the expiration time and to be alerted when the expiration date is approaching.</p> <p>See <i>Subscribe to Certificate Expiration Notices</i></p>
Cloud Build removed for SAP Mobile Cards	Deprecated	<p>Mobile Cloud Build has been removed from the Mobile Card Kit (MCK), and therefore removed from the SAP Mobile Cards user interface and documentation.</p> <p>See:</p> <ul style="list-style-type: none"> • <i>Configuring and Building Apps with the Cloud Build Service</i> • <i>Supported Build Types, Client Types, Packaging Details, and Build Options</i> • <i>Creating a Build Job</i>
Additional user interface languages	New	<p>The Mobile Services cockpit is now available in Serbian (Cyrillic), Serbian</p>

Features	Type of Change	Description
		<p>(Latin), Montenegrin, and Macedonian. The documentation is still translated only in Simplified Chinese and Japanese.</p> <p>See <i>User Interface</i></p>
New property added for <code>httpauth</code>	New	<p>The Enable User Attribute property has been added to the Security tab for HTTP settings. Roles can be derived from <code>httpauth</code>, similar to the way roles can be derived from SAML attributes.</p> <p>See <i>Configuring App Security</i></p>
Doc Update: Offline user interface improvements	Changed	<p>Additional information has been provided about using the <code>check_repeatable_requests</code> and also using any of the <code>send_standard_repeatability</code> or <code>send_origin_repeatability</code> properties.</p> <p>When the <code>check_repeatable_requests</code> property is enabled, both the <code>send_origin_repeatability</code> and the <code>send_standard_repeatability</code> properties are ineffective and should be disabled.</p> <p>See <i>Editing the Application Configuration File</i></p>
Supported versions for Cloud Build Service	Info only	<p>We have updated the supported versions information for the Cloud Build service.</p> <ul style="list-style-type: none"> March 2024 - Mobile Development Kit SDK version, iOS version, and notes February 2024 <ul style="list-style-type: none"> Mobile Development Kit SDK version, Android and iOS versions, and notes

Features	Type of Change	Description
		<ul style="list-style-type: none"> SAP Asset Manager SDK version, Android and iOS versions, and notes <p>For the latest information, see <i>Supported Build Types, Client Types, Packaging Details, and Build Options</i>.</p>

Related Information

[User Interface \[page 39\]](#)

[Creating a Destination \[page 92\]](#)

[Testing OData Destination Quality \[page 112\]](#)

[Editing the Application Configuration File \[page 122\]](#)

[Subscribe to Certificate Expiration Notices \[page 264\]](#)

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

[Creating a Build Job \[page 168\]](#)

[Configuring App Security \[page 282\]](#)

[Supported Build Types, Client Types, Packaging Details, and Build Options \[page 161\]](#)

1.4 Get Started

SAP Mobile Services offers authentication, secure onboarding, native push notifications, and reporting capabilities for enterprise mobile applications. Mobile Services cockpit provides a single comprehensive Web administration and monitoring portal for configuring and managing mobile applications.

In Mobile Services cockpit, set up developer and customer accounts.

Note

Mobile Services is available both stand-alone and as part of SAP Build Code. If you plan to use SAP Build Code, see [What is SAP Build Code](#) to learn more about the service.

If you are using this service as part of SAP Build Code, follow the [SAP Build Code Initial Setup](#) instructions.

[Enable Developer Accounts \[page 19\]](#)

A developer account allows you to explore the basic functionality of SAP Mobile Services. A developer account is also called a trial account, and access is open to everyone.

[Set Up Customer Accounts \[page 20\]](#)

An SAP Mobile Services customer account allows you to host production, business-critical applications that are supported 24/7. A customer account is also called a production account.

[Using Mobile Services Cockpit in an Extended Landscape \[page 29\]](#)

You can access extended Cloud Foundry landscapes from Mobile Services cockpit using the API Endpoint of an extended landscape.

Related Information

[Migrate: SAP Mobile Services Neo to Cloud Foundry \[page 530\]](#)

[Importing Application Configurations \[page 228\]](#)

1.4.1 Enable Developer Accounts

A developer account allows you to explore the basic functionality of SAP Mobile Services. A developer account is also called a trial account, and access is open to everyone.

Context

As a Mobile Services developer, you may wish to test your applications in an end-to-end environment. A developer trial account gives you access to administrative features to help you do that. To learn more about the developer trial account, see [Getting a Global Account](#) and [Getting Started with a Trial Account in the Cloud Foundry Environment](#).

Note

Please be aware that trial accounts are restricted in usage. See *Service Plans* for information about plans, capabilities, and limitations,

The Mobile Services enablement for developer trial accounts is identical to the procedure in *Set Up Customer Accounts*.

When you select a space and log in, Mobile Services cockpit performs a health check and informs you of configuration problem such as:

- Missing core services
- Missing mobile services
- Exceeding instance limit threshold
- Exceeding memory limit threshold

This enables you to figure out setup problems right away.

Related Information

[Service Plans \[page 23\]](#)

1.4.2 Set Up Customer Accounts

An SAP Mobile Services customer account allows you to host production, business-critical applications that are supported 24/7. A customer account is also called a production account.

Prerequisites

To explore and use the powerful capabilities of Mobile Services cockpit, verify that you:

- Have purchased a license for SAP Mobile Services.
- A single user within your organization should have access to the global account, and is known as an administrator. This user has full access to SAP BTP cockpit and can assign subscriptions for Mobile Services cockpit to your organization's SAP BTP account.
- Have one global account and at least one subaccount in the Cloud Foundry infrastructure. See [Getting Started with an Enterprise Account in the Cloud Foundry Environment](#) to verify.
- Know whether Feature Set A or Feature Set B is assigned to your global account. To find out, right click your user name in the upper right corner of SAP BTP cockpit, and select [About](#). The [Cloud Management Tools](#) property identifies your assigned feature set. To learn more about feature sets, see [Cloud Management Tools – Feature Set Overview](#).
- Have basic knowledge of Cloud Foundry entitlements and quotas, and Mobile Services quotas. See [Mobile Services Quotas](#).

Note

Do not change any of these settings.

Context

To set up the Mobile Services cockpit and integrate it into your internal landscape, a SAP BTP account administrator must perform the following steps.

Procedure

1. Log into your Global Account, and navigate to the menu option [Subaccount Assignments](#) under [Entitlements](#).
2. Select the Subaccount for which you want to enable Mobile Services. If you haven't yet created a Subaccount, go to the menu option [Subaccounts](#) and create a new account first.
3. Click on [Configure Entitlements](#) and select [Add Service Plan](#).
4. Select [Cloud Foundry Runtime](#) and [Mobile Services](#). Depending on your license, Mobile Services might show up with two different service plans, "b2c" or "standard". Even though you can use both from one Subaccount, please only select the one you would like to use. The plans are charged for differently. Please familiarize yourself with the price plan before using the service plans.

5. Set the [Subaccount Assignment](#) for "Mobile Services" to 1 unit.
6. Determine whether you need to assign additional Cloud Foundry Runtime to the Subaccount. Mobile Services needs at least 1GB (smallest unit) of Cloud Foundry Runtime assigned to the Subaccount. Cloud Foundry Runtime also controls routes and services instances. See *Mobile Services Quotas* for additional information.
7. If you have not yet done so, create a Space and add Space Members using information provided in [Managing Spaces](#). You'll need to be aware of whether your account uses Feature Set A or Feature Set B.

Once the Space is created and set up, navigate to the Space and click on [Service Marketplace](#) in the left menu. A tile for "Mobile Services" should appear. If this is not the case then restart and ensure the right Subaccount is selected.

Note

The SAP BTP Cloud Foundry Space Manager and Org Manager can manage space members and their roles. For more information about Cloud Foundry default roles, see [Orgs, Spaces, Roles, and Permissions](#). Mobile Services cockpit read / write access is controlled by these roles:

- Space Developer role – read and write access to Mobile Services.
- Space Auditor role – read access to Mobile Services.
- Space Manager role – read access to Mobile Services.
- Org Manager – read access to Mobile Services.
- Space Supporter – read access to Mobile Services

The Space Manager and Org Manager roles have read-only access in the Mobile Services context, with the exception of being able to assign the Space Developer role to any user.

8. To access Mobile Services cockpit, select Mobile Services in the Marketplace and click on the [Support](#) link shown in the service details. Or, create a service instance or use an existing service instance, and select the [Open Dashboard](#) link shown for the service instance.

Note

If you do not find the Mobile Services option listed, this may indicate that the Cloud Foundry landscape is an extended landscape, since extended landscapes do not appear in the [Support](#) link. To use Mobile Services cockpit in an extended Cloud Foundry landscape, you must create a mobile services instance from SAP BTP cockpit. To access the Mobile Services cockpit, select the mobile services instance link, [Open Dashboard](#). For more information see *Using Mobile Services Cockpit in an Extended Landscape*.

See the complete list of SAP UI5 supported browsers on [SAP Help Portal](#).

If a Mobile Services cockpit session times out and displays a blank screen, refresh or restart the browser.

9. (Optional) To access on-premise back ends, install and configure SAP cloud connector. See [Installation](#).

Note

In Mobile Services cockpit, configure on-premise HTTPS back-end connections as HTTP using the virtual host address. Communication between the cloud and your on-premise SAP cloud connector is secure. Communication between SAP cloud connector and your back-end system uses standard HTTPS security.

- a. In SAP cloud connector, verify that necessary back-end service URLs are allow-listed. Every on-premise URL that is configured in Mobile Services cockpit, such as application endpoints or the security configuration, must be allow-listed. See [Configure Access Control \(HTTP\)](#).
 - b. Generate a system certificate and import it into SAP cloud connector. See [Initial Configuration \(HTTP\)](#).
10. Access one of the Mobile Services cockpit URLs described in step 8.

Note

When you select a space and log in, Mobile Services cockpit performs a health check and informs you of configuration problem such as:

- Missing core services
- Missing mobile services
- Exceeding instance limit threshold
- Exceeding memory limit threshold

This enables you to better figure out setup problems right away. If a warning appears, return to the [Entitlements](#) section and try to fix the issue. Ignoring the warnings can lead to unexpected results.

11. Create an application in Mobile Services cockpit. See [Configuring Applications](#).

[Mobile Services Quotas \[page 22\]](#)

Describes how Mobile Services consumes Cloud Foundry Runtime.

[Service Plans \[page 23\]](#)

Describes the service plans that are available for Mobile Services.

Related Information

[Troubleshooting: Common Issues \[page 318\]](#)

[Configuring Applications \[page 45\]](#)

[Native Push Notification for a Back End \[page 354\]](#)

[REST API Application Development Overview \[page 323\]](#)

1.4.2.1 Mobile Services Quotas

Describes how Mobile Services consumes Cloud Foundry Runtime.

Use this information to analyze your requirements, and configure your quota plan for Mobile Services.

Basic Consumption

Mobile Services creates one route and two service instances in a customer Space. Mobile Services also creates some additional routes and service instances for system mobile apps (such as SAP Mobile Cards and AppLab),

and one Cloud Foundry application, `BindMobileApplicationToMe`, to which all routes are bound. Each of the system mobile apps holds one route and two service instances.

In summary, to create N mobile apps in Mobile Services requires:

- 128M memory
- N+3 app routes
- 2 * (N+3) service instances

In the Cloud Foundry environment, a quota of memory, routes and service instances is assigned per Cloud Foundry runtime unit. One runtime unit is typically 1G memory + 10 routes + 100 service instances. (For some old subaccounts, one runtime unit is 1G memory + 10 routes + 10 service instances). Please refer to the platform documentation for information about mapping the application unit to memory, routes and service instances.

Consumption Example

If your account's Cloud Foundry runtime=1G memory+10 routes+100 service instances, you need the following to create N mobile apps:


Cloud Foundry runtime units = $\text{Ceil}(\text{Max}((N+3)/10, (N+3)*2/100, 128\text{M}/1\text{G}))$

For example:

- Units = 1 when N = 1, 2, 3, ... 7
- Units = 2 when N = 8, 9, 10, ... 17
- When N = 10, $\text{Max}(1.3, 0.26, 0.128) = 1.3 \sim 2$

1.4.2.2 Service Plans

Describes the service plans that are available for Mobile Services.

Use this information to analyze your requirements and select the right plan. The terms license type and service plan are used interchangeably. For detailed information, see [SAP Discovery Center](#)  (filtered for SAP Mobile Services).

Service Plan	Description	Comments
Free (free)	<p>Develop, configure, and manage mobile apps that access enterprise data. Free tier is available for production accounts and can be upgraded to a production plan. Try out a service using a global account without additional cost, and then move to a paid service once you reach the technical limits of the service. Only community support is available for free tier service plans, and these are not subject to SLAs. Additional restrictions may apply.</p> <p>See <i>Important Free Plan Information</i> and <i>Upgrading the Cloud Foundry Runtime</i> below for additional details.</p>	<p>No feature limitations, but quota limits apply:</p> <ul style="list-style-type: none"> • Maximum of 3 apps, up to 10 user registrations per app • Maximum destinations per app: 5 • Maximum concurrent requests per destination: 2 • Maximum request payload size: 1048576 bytes (compare 2G for Resources) • Maximum push registrations: 10 • Interval between two push notifications to the same registration: 2 minutes. • Maximum Mobile Cards registrations: 10 <p>Additional restrictions may apply.</p> <div> <p>Note</p> <p>Only community support is available for free tier service plans and these are not subject to SLAs.</p> </div>

Service Plan	Description	Comments
Lite (lite)	Develop, configure, and manage mobile apps that access enterprise data. The Lite plan is for mobile services in a trial landscape. Lite is only available for trial accounts and cannot be upgraded to a production plan. There are no feature limitations, but quota limits apply. Only community support is available for free tier service plans, and these are not subject to SLAs. Additional restrictions may apply.	<p>No feature limitations, but quota limits apply:</p> <ul style="list-style-type: none"> • Maximum of 10 apps, up to 3 user registrations each • Maximum destinations per app: 5 • Maximum concurrent requests per destination: 2 • Maximum request payload size: 1048576 bytes (compare 2G for Resources) • Maximum push registrations: 3 • Interval between two push notifications to the same registration: 2 minutes. • Maximum Mobile Cards registrations: 3 • Maximum Card Types: 5 • Maximum Cards per Card Type: 3 <p>Additional restrictions may apply.</p> <div> <p>Note</p> <p>Only community support is available for trial service plans and these are not subject to SLAs.</p> </div>

Service Plan	Description	Comments
Resources (Tiered Edition) (resources)	<p>Develop, configure, and manage mobile apps that access enterprise data. Key features include app content lifecycle management, push notifications for data updates, on-device storage of data (offline apps), app security, and app monitoring and usage reporting. Billing is based on active resources, which is a user identifier (or device identifier for anonymous users) that has accessed mobile services within the calendar month. No limitations are imposed on available features, resources, or apps.</p>	<p>No feature limitations. Unlimited resources and apps.</p> <p>Every day, resource identifier hashes are sent to our internal metering service for those that have accessed mobile services during the day. Each calendar month, cloud reporting and billing systems run distinction and aggregation processes for the resource identifier hashes to get the final number of active resources for the month. A resource is counted once if the user (or device in the case of anonymous users) uses the same name or name resource under multiple subaccounts.</p> <p>In fact, the main difference between Resources and Consumer Edition is pricing. Please refer to SAP Discovery Center for details (logon required).</p> <p>For example:</p> <ul style="list-style-type: none"> • If the application is authenticated explicitly (using a method such as OAuth, SAML, Basic, or HTTP) and is accessed by 100 users from 200 mobile devices in a month, it is counted as 100 resources. • If the application is accessed anonymously (using API-KEY) and is accessed by 100 users from 200 mobile devices in total in a month, it is counted as 200 resources. Mobile Services does not differentiate whether it's 100 actual users or 200 actual users.

Service Plan	Description	Comments
Build Code (build-code)	<p>This is a dedicated plan for the SAP Build Code. If you want to use this service with the SAP Build Code, use the "build-code" plan. Please note if you use the regular plan, it will incur charges outside the build code capacity units metric. Use this plan to develop, configure, and manage mobile apps for your enterprise.</p> <p>Enables customers who purchased Capacity Units for the Build Code Plan, to manage apps in Mobile Services.</p>	<p>No feature limitations.</p> <p>Mobile Services usage is converted to Capacity Units for Build Code.</p>
Kernel Service (kernel-service)	The license type that is used for an application that is managed in a single-app cockpit (Software as a Service, or SaaS). The license type is read-only and cannot be selected or edited in the cockpit.	Read only.
Users (standard) Deprecated	(Deprecated) Develop, configure, and manage mobile apps that access enterprise data. Standard plan billing is based on registered users. A registered user is a current named user that is associated with device registrations and application connections. A named user is an individual who is authorized to access the service. Users may be employees of customers or customer affiliates, or business partner users. There are no feature limitations.	<p>No feature limitations.</p> <p>At the beginning of each month, all registered user hashes are sent to our internal metering system. New registered user hashes are sent in the following days of the month. If a user is de-registered during the month, it is still counted in the current month and will not be counted in the next month. Users are considered registered until they de-register, whether they use the service during the month or not. A user is counted once if he or she uses the same user under multiple subaccounts.</p>

Service Plan	Description	Comments
Consumer Edition (b2c) Deprecated	(Deprecated) Develop, configure, and manage mobile apps that access enterprise data. Consumer Edition billing is based on active users. An active user is an individual user of a platform application that connects to the service at least once during any rolling three-month period. Billing is determined for active users that have accessed mobile services within the calendar month. There are no feature limitations, and unlimited users and apps are allowed.	<p>No feature limitations. Unlimited users and apps.</p> <p>Every day, user name hashes are sent to our internal metering service for every user who has accessed mobile services during the day. Each calendar month, cloud reporting and billing systems run distinction and aggregation processes for the user name hashes to get the final number of active users for the month. A user is counted once if he or she uses the same user under multiple subaccounts.</p>

Important Free Plan Information

When you create a mobile app in Mobile Services, Mobile Services creates a "BindMobileApplicationRoutesToME" Cloud Foundry app under the same space. When you enable the Cloud Foundry Runtime, you can choose a Cloud Foundry runtime plan that is either Free or Standard. If your Cloud Foundry Runtime is not using Free Plan, the "BindMobileApplicationRoutesToME" app incurs a cost.

Keep in mind:

1. If you want "totally" free, you should enable Cloud Foundry Runtime with the Free plan, and place Mobile Services in the Free plan's Org and Space.
The drawback of this selection is that when you choose to upgrade Cloud Foundry Runtime from the Free plan to a non-free plan, you must do so manually. If there are any other apps under this Free Cloud Foundry Runtime, they will begin to incur charges.
2. You can also use the Mobile Services Free plan under the Cloud Foundry Runtime Standard plan. Thus the runtime app does incur a charge. You can manually shut down the apps under the Org and Space to minimize the cost. The benefit is that you can convert the Mobile Services Free plan to non-free plan on the fly without upgrading Cloud Foundry Runtime.

Upgrading the Cloud Foundry Runtime

Following are the basic steps for upgrading the Cloud Foundry Runtime from Free to Standard plans:

1. From SAP BTP cockpit log into your Global Account and access [Cloud Foundry Runtime](#) and your [Subaccount](#).
2. Choose [Instances and Subscriptions](#) in the menu.
3. Scroll down to [Environments](#).
4. Click the menu button (...) and select [Update](#).
5. Change the plan from [Free](#) to [Standard](#) (non-reversible).

6. Click [Update Instance](#).

Related Information

[Enable Developer Accounts \[page 19\]](#)

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)

[Managing Application Features \[page 51\]](#)

[Configuring Assigned Features \[page 53\]](#)

[Configuring App Security \[page 282\]](#)

[Changing Service Plans at the App Level \[page 215\]](#)

1.4.3 Using Mobile Services Cockpit in an Extended Landscape

You can access extended Cloud Foundry landscapes from Mobile Services cockpit using the API Endpoint of an extended landscape.

Prerequisites

- An extended Cloud Foundry landscape must be in place.
- Know the API Endpoint of the Cloud Foundry organization you want to access.
- You can only access API endpoints in the same region.

Procedure

1. Open the Mobile Services cockpit using the [Service Marketplace](#) [Mobile Services](#) [Support link](#) . The Mobile Services cockpit dialog, [Select API Endpoint](#), appears.
2. In [API Endpoint](#), either select the default value, or enter the API Endpoint of the extended Cloud Foundry landscape. The API Endpoint can be found under [Cloud Foundry Environment](#) on SAP BTP cockpit, for example, `https://api.cf.us10-001.hana.ondemand.com`.
3. Select [Login](#) to access the URL.
4. After you log into the endpoint, the [Organization and Space](#) prompt appears. The current API Endpoint value appears.

You can choose the organization and space to access, and then select [Open](#) to access it. If the organization you want to access is not listed, you might have used the wrong API Endpoint. In this case, you can log out and log in again, in order to input the correct API Endpoint value.

Choose Organization and Space

Org/Space	Description
Organization	Select an organization, such as <code>mobile-tenant99</code> .
Space	Select a space, such as <code>dev</code> .

Once you have successfully logged into the organization and space, the associated URL appears in the [API Endpoint](#) list.

📘 Note

If you previously created a mobile services instance, you can still select its [Open Dashboard](#) link to open Mobile Services cockpit without using the API Endpoint.

1.5 Administration

Administrators interact with SAP Mobile Services to ensure the production environment works efficiently.

Administrator tasks consist primarily of configuring applications for deploying to users and monitoring application in the user community.

[SAP Mobile Services Overview \[page 31\]](#)

SAP Mobile Services provides services to mobile applications, such as application analytics, app resources, onboarding, HTTP/HTTPS configuration and so on.

[User Interface \[page 39\]](#)

Frequently used icons in the Mobile Services cockpit. Actual icon styles may vary slightly.

[Getting Support From the Cockpit \[page 42\]](#)

Get built-in support for Mobile Services without leaving the cockpit. You have access to blogs, articles, videos, and tutorials, and can use Expert Chat and create a support ticket if authorized.

[Accessibility \[page 43\]](#)

Government and public agencies that require adherence to Web Content Accessibility Guidelines (WCAG) can ensure that their employees with special needs have a productive experience with SAP Mobile Services.

[Application Administration \[page 44\]](#)

Use Mobile Services cockpit and other tools to manage and monitor native apps, cross-platform apps, web apps, and Mobile Development Kit clients. Managing includes defining and configuring applications; monitoring applications and application usage; viewing statistics and logs; checking system health; and troubleshooting problems.

[Security \[page 281\]](#)

Describes basic security features for SAP Mobile Services in the Cloud Foundry environment.

[Troubleshooting: Common Issues \[page 318\]](#)

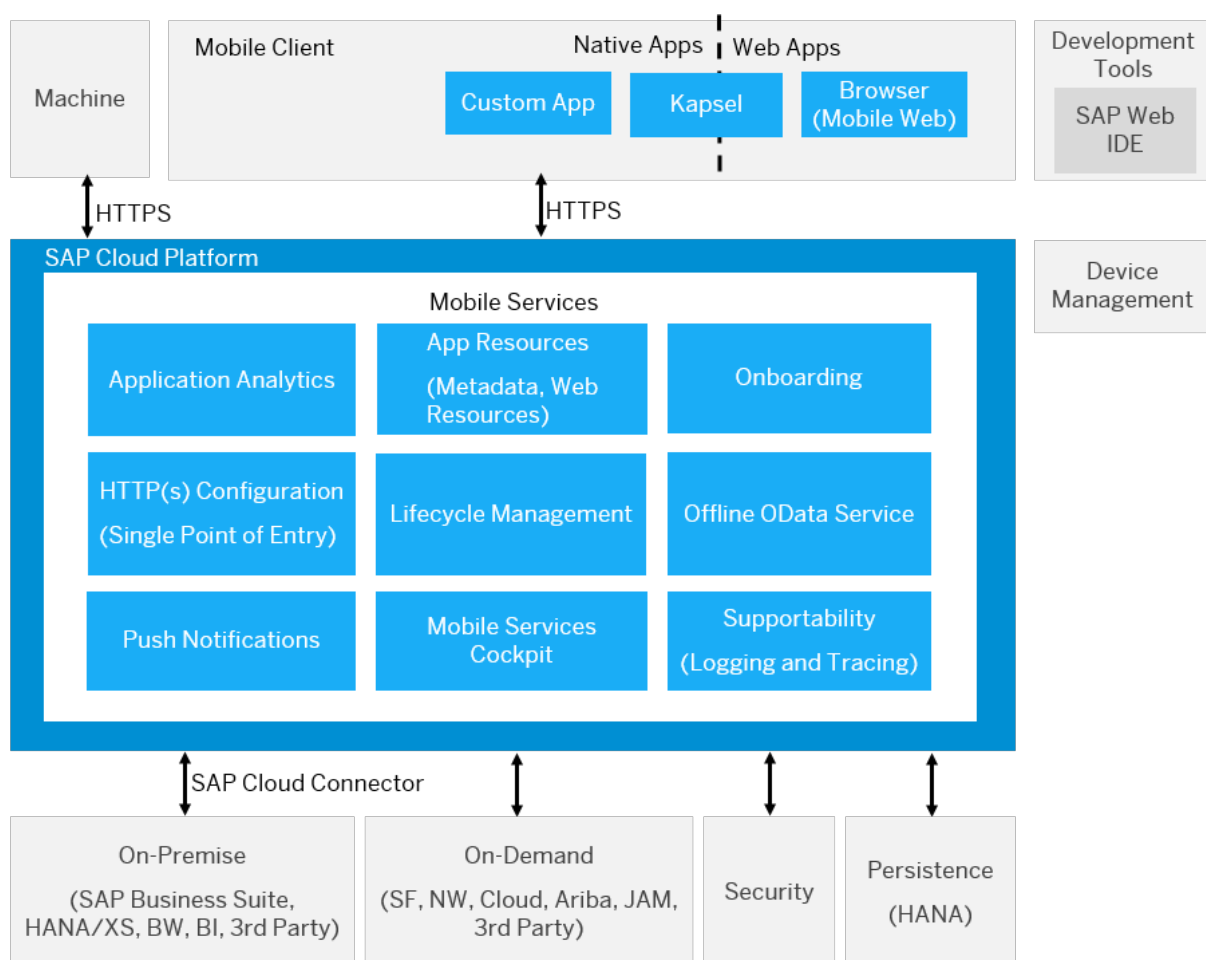
Overview of common issues.

Related Information

[Application Administration \[page 44\]](#)

1.5.1 SAP Mobile Services Overview

SAP Mobile Services provides services to mobile applications, such as application analytics, app resources, onboarding, HTTP/HTTPS configuration and so on.



Mobile application services consist of the following:

- Application Analytics – usage statistics that can be displayed graphically in Mobile Services cockpit.
- App Resources – containers of dynamic configurations, styles, or content that can be downloaded by native applications.
- Onboarding – authentication of users.
- Destination/Connectivity Configuration – open standards for client communications.
- Lifecycle Management – managing and deploying multiple versions of an application.
- Offline OData Service – optimizes data transport between the back end and the client offline store.

- Push Notifications – native notifications sent from back-end systems to the server, which forwards them on to the clients.
- Mobile Services cockpit – deploying, managing, and monitoring applications.
- Supportability – logs for monitoring mobile application health and troubleshooting.

Mobile Services can expose on-premise back-end services through SAP Cloud Connector, and on-demand back-end services directly.

SAP BTP security enables you to use an on-premise identity management system for on-demand applications.

[Logging Overview \[page 32\]](#)

SAP Mobile Services includes logs and traces that enable administrators, developers, and support professionals to troubleshoot application issues.

[Application Usage Reporting Overview \[page 33\]](#)

You can collect standard usage information for applications, and view reports based on information logged by clients and uploaded to the server.

[Offline Applications Overview \[page 34\]](#)

Offline support enables client applications to access back-end data without requiring a constant online connection to the back end.

[Discovery Service Overview \[page 35\]](#)

(Not available in all COUNTRY/REGIONS) The SAP Discovery Service lets you distribute initial configuration data to a mobile app and enhances the user onboarding experience.

[Push Overview \[page 36\]](#)

Use the push feature to push updates from the back-end data source to applications that are running on mobile devices. The back end can also push notifications to apps that provide a certain capability or run on a specific device type, rather than to particular applications.

[Hybrid App Feature Restriction Overview \[page 38\]](#)

You can set feature restriction policies for hybrid apps in Mobile Services cockpit. This gives you additional control over the SAP Fiori Client application features that are allowed and restricted.

[Offline Store Upload Overview \[page 38\]](#)

End users can securely upload local database files from a supported device to the server for analysis.

1.5.1.1 Logging Overview

SAP Mobile Services includes logs and traces that enable administrators, developers, and support professionals to troubleshoot application issues.

Logs fall into two categories:

Event Logs	Generated by the server and provide information about the activities of mobile application users. By default only warnings and errors are logged. These contain information about any problems that a mobile application user has experienced. This could be a problem with requesting data from a back end, for example. To also view success messages, enable them for each application and assigned feature. See Viewing Event Logs [page 237] for additional information.
Client Logs	Generated by mobile applications and uploaded to the server. These can be enabled for each mobile application by activating the Mobile Client Log Upload feature (see for additional

[information](#)) [\[page 58\]](#). Based on the configuration, mobile applications can upload log files to the server and an administrator can download them for problem investigation.

Related Information

[Viewing Event Logs \[page 237\]](#)

1.5.1.2 Application Usage Reporting Overview

You can collect standard usage information for applications, and view reports based on information logged by clients and uploaded to the server.

All records collected from the device are tagged with the following attributes:

- Application: application bundle ID and version
- Device and operating system: operating system platform, platform version and device model name
- User sessions: an instance of application running in the foreground

The administrator has complete control over the usage reports upload in the SAP Mobile Services, and can view reports and carry out necessary operations.

Enable Usage Collection for an Application

The administrator can configure the uploaded records in the server remotely using Mobile Services cockpit and optimize them using WiFi. This process minimizes the impact of usage collection on the end user's cellular data plan.

Enable Application-Specific Columns in the Database

The administrator can enable developer-defined usage report collection in Mobile Services cockpit, and enable or disable the creation of application specific columns in the database on per-application basis.

The application developer must include a reporting library where a standard set of information is captured for every application. If developers have developed custom information to be logged, you can collect that information as well.

Set the Maximum Threshold for Storing Records

The administrator can set the maximum number of client records to be stored on devices. On exceeding this limit, data is uploaded over cellular data.

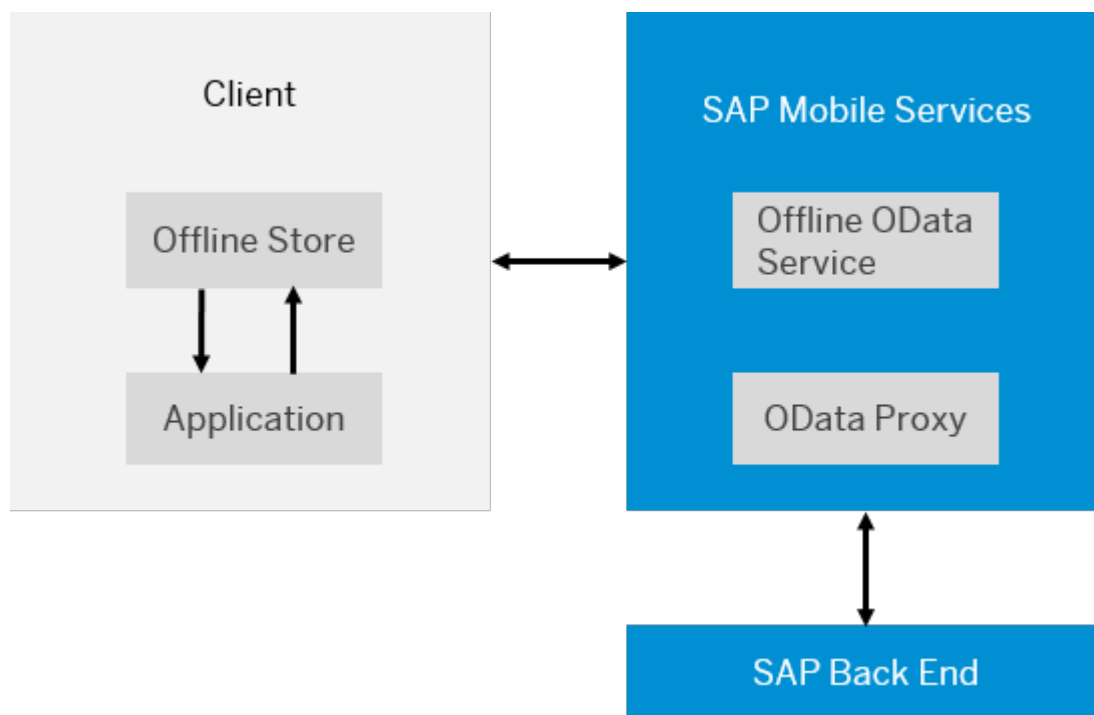
1.5.1.3 Offline Applications Overview

Offline support enables client applications to access back-end data without requiring a constant online connection to the back end.

You might want to run applications offline to:

- Improve performance by accessing offline data instead of sending data requests to SAP Mobile Services.
- Enable users to continue to use applications when there is intermittent network coverage.
- Support business processes that must be executed by a user while the application is offline.

To work offline, an application must initialize an offline store, which stores data that the application can access when it is offline. SAP Mobile Services provides an Offline OData Service that moves data between the back end and the client offline store.



SAP Mobile Services retrieves data from an OData service that is running in a back end, and from that data creates an initial database on the client. On an ongoing basis, SAP Mobile Services updates the client database based on changes, or deltas, that have been made to the data on the back end. Deltas between the back-end data and the client data are identified either by the back end or by SAP Mobile Services.

You can configure offline applications to optimize offline performance by defining:

- Column indexes for the client database

- Common user data to cache on the server to reduce the amount of data that needs to be synchronized with the back end.

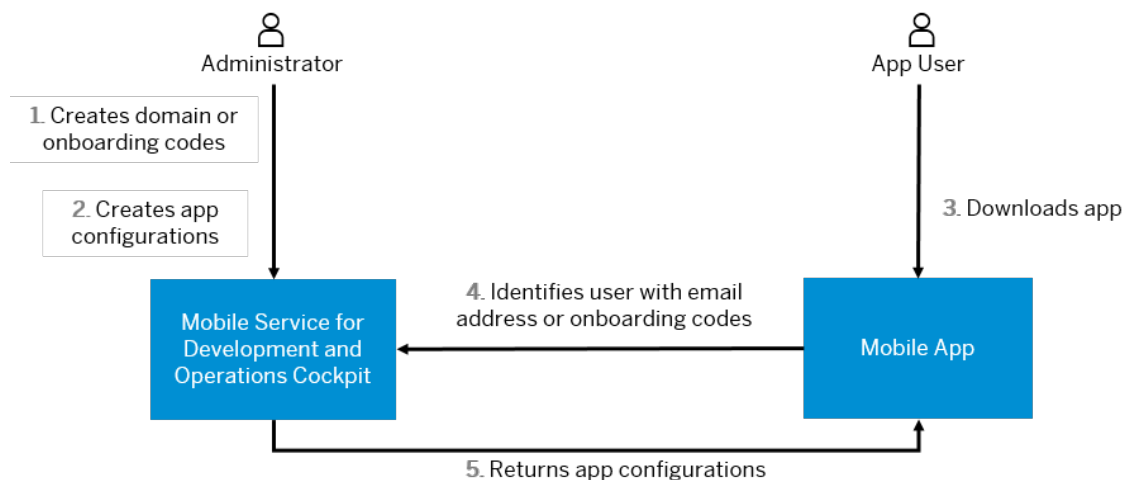
When an application is offline, it accesses data from the offline store. Any updates that are made while the client is offline are stored locally and become pending updates for the back end. When the client comes back online, SAP Mobile Services updates the back end by processing the pending updates.

Related Information

[Defining Offline Settings for Applications \[page 119\]](#)

1.5.1.4 Discovery Service Overview

(Not available in all COUNTRY/REGIONS) The SAP Discovery Service lets you distribute initial configuration data to a mobile app and enhances the user onboarding experience.



Ensure that the application developer has added the Discovery Service procedure to the application. See these sections for the procedure:

- [App Configuration Using a Discovery Service Provider.](#)

After the Discovery Service procedure is added to the application, use the Mobile Services cockpit to add the application configurations to the app itself.

Related Information

[Enabling Applications to Discover Configurations \[page 179\]](#)

1.5.1.5 Push Overview

Use the push feature to push updates from the back-end data source to applications that are running on mobile devices. The back end can also push notifications to apps that provide a certain capability or run on a specific device type, rather than to particular applications.

You can use SAP Mobile Services to manage push for individual applications that use native notifications.

Developers enable native push notification in the application code, and link the certificate with the mobile application at build time. Users download the application from a market place, such as Apple App Store, Google Play, or similar service, and, when a change occurs in the back end, it sends a push notification to mobile applications on devices with push enabled.

Push Notification

For native mobile applications, the platform manages the certificates, tokens, and push notifications for individual applications. When changes occur, the back end sends push notifications to mobile applications on devices that are push enabled.

Push Desk Notification

For native and hybrid applications, Administrators can send push notifications to all users of a push-enabled application from the Mobile Services cockpit. You can check all registered users and devices and select an individual user to whom to send a push message. This functionality is mostly used to test the end-to-end setup and configuration of the push functionality.

Predefined Push Configuration

You can enable or disable preconfigured push settings for applications. When enabled, the default push configuration that comes with the Apple App Store and Google Play version of the app is used. When disabled, you can configure push settings.

Custom Push

The Custom Push service enables Mobile Services to send push notifications to a custom push server for distribution. Your custom push server must implement the Custom Push API, which is able to receive push notification messages from Mobile Services, and coordinate distribution of push notifications to mobile clients. To configure this feature in Mobile Services cockpit, an administrative user must create a destination that points to the custom push server URL, and configure "Custom Push" for the application using the destination, as described in *Custom Push*.

Push Statistics

You can find statistics about push messages in the server data analytics section (see *Analytic Charts > Viewing Server Data Charts*).

Capabilities-based Push Support

Capabilities-based push enables a back-end to trigger a push to applications that provide a certain capability. Developers configure application connections to handle capabilities using the REST API (see *REST API Application Development Overview*). Devices send capability type information during registration or update. SAP Mobile Services maintains the mapping between capabilities and applications.

Device-type (form factor) Support

Devices send device type information to the server during registration. Device types are categorized into groups using the form factor property. The client can use any non-empty string for the device type (case insensitive), such as SmartPhone, phone, Watch, desktop, and so forth.

Note

Hybrid apps do not support device-type (form factor), with the exception of Fiori Clients downloaded from the App Store, and custom Fiori Clients.

Configuration of Capabilities

Application capabilities are part of the central application connection configuration. Similar to how the administrator controls some device capabilities from the server through feature policies, users control some application capabilities from the device. Device capabilities are controlled by the application and sent to the server. The capabilities are exchanged between the mobile app as part of the registration and settings exchange.

Through that mechanism, the mobile app can also override default capabilities. This gives users more control, enabling them to turn off certain capabilities for a mobile app instance, which translates into turning off native push notifications for a certain action into a particular application. Push can still be offered, but at the capability level, rather than individual application level.

Related Information

[REST API Application Development Overview \[page 323\]](#)

[Custom Push \[page 147\]](#)

1.5.1.6 Hybrid App Feature Restriction Overview

You can set feature restriction policies for hybrid apps in Mobile Services cockpit. This gives you additional control over the SAP Fiori Client application features that are allowed and restricted.

When you edit a hybrid app in Mobile Services cockpit, available feature plugins are listed on the [Mobile Settings Exchange](#) screen. You can indicate features that should be restricted from the user. Feature plugins are typically JavaScript APIs that provide access to the native APIs of the mobile device (implemented as Apache Cordova plugins). Plugins include:

- Cordova Camera
- Barcode Scanner (plugin for different types of barcode scanners, using the device's camera)
- Cordova Contacts
- Cordova File
- Cordova Geolocation
- Cordova Calendar
- Print
- SAP Push Plugin

You can also manage a list of feature restriction policies for all applications from a central location. Each of the centrally maintained feature restriction policies work as a template. An updated template is automatically applied to new hybrid applications, and can be manually applied to existing ones. Override the template for individual hybrid apps.

1.5.1.7 Offline Store Upload Overview

End users can securely upload local database files from a supported device to the server for analysis.

The administrator enables database upload for a specific application via the cockpit. The developer uses an API to enable client upload capability in specific applications, enabling later access to the uploaded database files to troubleshoot problems. The server checks for offline store files upon start up, and every hour, and automatically deletes expired files.

Developers (Application Development)

The developer uses the Database Upload API to enable client upload capability in a specific application. Typically a prompt is provided for the application user to upload a file for analysis. By default the maximum file size is 32 MB.

Administrators

The administrator enables the database upload policy for a specific application via the cockpit. The policy identifies how much time elapses before the uploaded database files are deleted automatically, and the maximum database file size that can be uploaded (required only if greater than the default of 32 MB). Allow ample time for the developer to review the uploaded files.

Device Users

When prompted, an application user uploads a file from the device for analysis.



Developers (Troubleshoot)




The developer analyzes the database file to solve a problem encountered by the user. The file is deleted from the server when the policy time limit has expired.













1.5.2 User Interface






Frequently used icons in the Mobile Services cockpit. Actual icon styles may vary slightly.

Mobile Services cockpit Icons

Icon	Purpose	Description
	<i>Toggle Menu</i>	Toggle between showing and hiding the navigation menu in the left pane. Display the menu for fast, easy navigation; hide it to maximize the window area.
	<i>Home</i>	<p>The Mobile Services cockpit home screen shows the last five apps that were accessed, a summary of alerts from the last 24 hours, and relevant KPIs for new registrations and push messages.</p> <p>The title indicates "Preview" if you are accessing the preview system, to distinguish it from the production system. Preview enables you to test new Mobile Services features for a week or so before they are released for production, but the preview landscape is not intended as a development or production environment.</p> <p>Your login name appears in the upper right corner. Select your name to view a drop-down menu. If <i>Helpdesk</i> appears after your name, this indicates that you are in read-only mode. Some buttons and features will be grayed out or not available.</p>

Icon	Purpose	Description
	About	<p>View information about the platform. Click your login name in the upper-right corner to see the option.</p> <ul style="list-style-type: none"> • Build: The date and time of the software build, typically in the format YYYYMMDD–HHMM. For example: 20230114–0832 indicates that the build was made on January 14, 2023 at 08:32 AM. • Version: The Mobile Services software version. The format YYMM indicates the general availability year and month, such as 2301. The format PLxx indicates a patch version, such as PL10.
	Settings	<p>Select your login name to view the drop-down menu. Select Settings, configure your settings, and then Save.</p> <ul style="list-style-type: none"> • Theme: select your preferred color and contrast option from the list, which includes Default, High Contrast Black, or High Contrast White. The Mobile Services cockpit user interface reloads in the new theme. • Language: select the language to use for the Mobile Services cockpit. The user interface reloads in the new language. For supported languages, such as Arabic, the primary calendar is reloaded in right-to-left format and the Gregorian calendar is used as secondary. Note that business logic, such as calculating by periods and running business processes, is still based on the Gregorian calendar • Time Zone: select either the local or UTC time zone format. Reports are also displayed in local or UTC format, depending on your selection.
	CMK Information	<p>(Restricted Availability) Check the status of the keys from SAP Data Custodian Key Management Service used by Mobile Services cockpit. Statuses include:</p> <ul style="list-style-type: none"> • Not Enabled - everything is stable in the Mobile Services cockpit but there is no place to check customer-managed key status. • Pending - the customer-managed key environment is not yet enabled. You may receive an alert saying that all Mobile Services cockpit interactions are blocked. You must finish all customer-managed key related configurations before accessing Mobile Services cockpit. • Enabled - everything is stable in the Mobile Services cockpit and you can check the status in the profile menu list. • Revoked - all Mobile Services cockpit interactions are blocked. After you revoke Customer-managed keys, Mobile Services cockpit cannot be used anymore. The next step is to delete the subaccount.
	Notifications	<p>Select to view notifications. For example, you may be alerted if a service plan limit has been reached or if you should check log files for information. Notification may provide useful information when configuring and using Mobile Services.</p>

Icon	Purpose	Description
	Give Feedback	Respond to a short survey and provide free-form text. Your feedback is anonymous and used to improve SAP Mobile Services. No personal data is collected. The survey changes periodically and currently is in English only.
	Support	Select to access built-in support without leaving the Mobile Services cockpit. You have access to blogs, articles, videos, and tutorials. You can also create support tickets and use Expert Chat, if you are authorized and sign in with your SAP Universal ID.
	Help	Toggle embedded help on and off. You can expand and collapse the Help Topics pane to view and hide the topics while you work. If a Guided Tours icon appears, select it and then the guided tour name.
	Guided Tours	Guided Tours walk you through the basic steps of a procedure from within the Mobile Services cockpit, such as configuring the Mobile Network Trace feature. <ol style="list-style-type: none"> 1. Select the help icon (). If a Guided Tour is available, the icon appears. 2. Select the Guided Tour icon. One or more Guided Tours may be listed. 3. Select the Guided Tour (this is required even if only one is available). 4. The first Guided Tour dialogue appears. Follow the instructions.
<div>  Note Behavior may differ based on screen type and size. If necessary, you can drag a dialogue to a new location to see text or user interface elements. </div>		
	Log Out	Log out of the application. Click your name in the upper-right corner to see the option.
	Create	Configure a new resource for a feature, such as a new destination.
	Action Settings	View available actions, such as customizing table columns.
	New	Add a new item, for example, a destination, a provider, or a feature restriction policy.
	Ping	Ping a destination.
	OData Application Destination Test Launch Web App	Depending on the context: <ul style="list-style-type: none"> • When setting up a destination, test an OData application destination. • When working with the ESPM sample data in Mobile Services cockpit, launch a simple Web user interface to test your app.
	Launch in Browser	Test a destination for scenarios that do not work from Mobile Services cockpit, because they require a business user (end user) login.

Icon	Purpose	Description
	Sort	Sort a list based on criteria you choose, such as ascending or descending order, or the column name.
	Upload	Upload an object.
	Download	Download an object.
	Details	See details about the selected resource.
	Copy URL	Copies a URL string to the clipboard, enabling you to paste the complete URL elsewhere.

1.5.3 Getting Support From the Cockpit

Get built-in support for Mobile Services without leaving the cockpit. You have access to blogs, articles, videos, and tutorials, and can use Expert Chat and create a support ticket if authorized.


Prerequisites


If you plan to create support tickets and use Expert Chat, you must sign in with your SAP Universal ID. If you are not authorized for these activities, these options are not available.

Procedure

1. In Mobile Services cockpit, select the headset icon  to access embedded support.

The first time you access embedded support, the [Welcome to Built-In Support](#) screen appears with onboarding screens to help you get started. Select [Next](#) to view the screens and then [Get Started](#) to access the [Built-In Support](#) screen.

Alternatively, you can select [Skip](#) to bypass the onboarding screens and go directly to the [Built-In Support](#) page. You can revisit the onboarding screens at any time by selecting  from the [Built-In Support](#) screen.

2. From the [Built-In Support](#) screen you can do the following:
 - Enter a topic and select the search icon . For example, enter `logs` or `xsuaa` to search for information about the topic.
 - Under [Our Recommendations for This Page](#), access useful links such as the [What's New for SAP Business Technology Platform](#), information about managing apps from the Mobiles Services cockpit, the SAP community, and the Mobiles Services product page.
 - From the banner:

- Select ⓘ to access the onboarding screens, watch video tutorials and access help for built-in support, set cookie preferences, and more.
 - Select 👤 to sign in with your SAP Universal ID and access your system context information.
 - Select ✕ to close the *Built-In Support* screen.
3. Select *Sign In* to sign in with your SAP Universal ID to get access to built-in support functions like case creation and Expert Chat.

You are informed if no S-user is authorized and prompted to link your existing accounts like S/P user IDs to get access to all built-in support options. If you are eligible, you can follow the prompting. To learn more, see [SAP Note 2617792](#) - Digital Support Experience Troubleshooting Guide (login required).

1.5.4 Accessibility

Government and public agencies that require adherence to Web Content Accessibility Guidelines (WCAG) can ensure that their employees with special needs have a productive experience with SAP Mobile Services.

This section describes how Mobile Services provides a productive experience.

Accessibility (Cloud Foundry)

Feature or function	Description
Alt tags and keyboard entries	Screen readers can read the alt tags added to pages, buttons, links, menus, navigation regions, light boxes, and field entry elements. They also announce keyboard actions as they are performed in Mobile Services, for example, <code>Tab</code> , <code>shift tab</code> , or <code>enter</code> . Page headings, column headings, button names, and links are also read by the screen reader.
Calendars with events	The calendar can be read by screen readers. Day cells on the calendar date picker are read as the full date.
Color contrast	Page elements can display in high contrast mode. Each user selects their own theme from <i>Settings</i> , including the default, high contrast black or high contrast white. The user interface is reset in the selected theme.
Edit button in table row	The edit action is correctly identified by screen readers.
Input and edit messages	When an input or edit message prompts you to enter information, the screen reader reads the message prompt.
Mobile Services cockpit homepage and heading structure	Standard heading elements in HTML are used for the Mobile Services cockpit homepage and heading structure to facilitate screen reader navigation.
HTML header elements	On the company home page, the Mobile Services cockpit homepage, and its subsequent pages, headers are recognized by screen readers.
Image tool tips	All icons and images in Mobile Services have tooltips that appear when you hover the cursor over them. Tooltips are also read out by the screen reader.

Feature or function	Description
Item count	The item count for a menu's listed items are read.
Keyboard controls	A tab sequence supports keyboard tab actions to navigate through menus, jump to pages (for example, buttons, links, text boxes, and other fields with the option or requirement for data entry or data selection).
Language attributes	Supported translated languages can be selected from a list in Mobile Services cockpit. Each user selects their own language option from Settings , and the user interface is reset in the selected language. System messages are also translated.
	<div> <div>📌 Note</div> <div>Documentation is available in English, Simplified Chinese and Japanese.</div> </div>
Page element focus	A visual indicator displays when the cursor focus is placed on page elements (for example, buttons, text boxes, or checkboxes).
Processing action	When you are waiting for an action to complete and there is a visual representation of the processing, the screen reader explains that the action is in progress.
Profile pages	Keyboard support for all page elements using <code>Tab</code> and <code>enter</code> , <code>arrow</code> , or <code>escape</code> keys. All page elements are identified with labels. All table cells are identified in the order of value, title, row, and column.
Tab chain	You can use the keyboard controls to reset the tab focus for a given element. For example, you can use <code>Ctrl</code> + <code>L</code> to change the location focus or <code>Alt</code> + <code>D</code> to move to the search box.
Tables	When you tab to a table cell, the screen reader reads its location and content by row number, column title, and column number.

1.5.5 Application Administration

Use Mobile Services cockpit and other tools to manage and monitor native apps, cross-platform apps, web apps, and Mobile Development Kit clients. Managing includes defining and configuring applications; monitoring applications and application usage; viewing statistics and logs; checking system health; and troubleshooting problems.

Native (online and offline), hybrid (Kapsel - offline), and Web applications are developed using a variety of tools and methods. SAP tools facilitate the development of mobile apps, with modularized methods for downloading, logging on, push notification, and error reporting. During the development process, a unique application identifier is generated for each application, and the application is deployed to an application download site or to SAP Mobile Services. Web applications are running on-premise, but securely exposed through SAP Mobile Platform or SAP Mobile Services.

The administrator creates an application definition in Mobile Services cockpit, which includes the unique application identifier, plus the connection to its backend data source in the production system, the security configuration, and application-specific entries.

The administrator provisions applications to devices through native application stores, or through enterprise Web site downloads. When a user logs in to an application (or accesses the application as an anonymous user), the application+user+device combination is registered in Mobile Services cockpit. This registration enables you to manage and monitor device applications in the field using Mobile Services cockpit, and to take advantage of individual and aggregate usage statistics.

[Configuring Applications \[page 45\]](#)

Create an application definition that enables you to manage the application using Mobile Services cockpit. The application definition includes a unique application identifier, connections to the back-end data source, and optionally, other feature settings.

[Enabling Applications to Discover Configurations \[page 179\]](#)

(Not available in all COUNTRY/REGIONS) Using Mobile Services cockpit, you can add the application configurations or link the domain from the app itself. You can update or delete the configurations at any time.

[Managing SAP Build Apps \[page 183\]](#)

Configure and manage an SAP Build App in Mobile Services cockpit.

[Configuring Micro Apps \[page 186\]](#)

[Managing and Monitoring Applications \[page 209\]](#)

Use Mobile Services cockpit to manage applications, registrations, users, back-end connections to the data source; view application usage statistics; and manage and view application reports.

1.5.5.1 Configuring Applications

Create an application definition that enables you to manage the application using Mobile Services cockpit. The application definition includes a unique application identifier, connections to the back-end data source, and optionally, other feature settings.

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

[Managing Application Features \[page 51\]](#)

You can add features to an application from the Features menu option. Features enable capabilities that developers have added to an application.

[Configuring Assigned Features \[page 53\]](#)

Configure features associated with an application definition.

Related Information

[Set Up Customer Accounts \[page 20\]](#)

[Migrating to SAP Mobile Services \[page 534\]](#)

[Configuring App Security \[page 282\]](#)

1.5.5.1.1 Defining Applications (Cloud Foundry)

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

Context

When you define an application, choose a configuration template. Keep in mind:

- When you select a Native, Hybrid, or Mobile Development Kit template, the most typical features for the application type are selected automatically. You can add or remove features at this time, or take care of it later.
- Some features, such as Mobile Settings Exchange, are required, so you cannot remove them.
- Other features have dependencies, so you either cannot remove them, or must remove them in a particular order. For example, Mobile Offline Access requires Connectivity, and Mobile Cloud Build requires Mobile App Catalog.

After you define the application, configure the assigned features, or add more features. See *Managing Application Features* and *Configuring Assigned Features*.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) , and click [New](#).

Note that some service plans impose a limit on the number of applications that you can implement. When you reach the limit, a message appears, such as "The current activity cannot be performed due to limitations on the current plan". To mitigate, delete one of the applications. See *Service Plans* for information about service plan limits.

2. In [New Application](#), on the [Basic Info](#) page, enter the application properties, followed by [Next](#):

Application Properties

Field	Value
ID	<p>Unique identifier for the application, in reverse-domain notation. This is the application or bundled identifier that is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Mobile Services, and client applications use the application ID when sending requests to the server.</p> <p>An application ID:</p> <ul style="list-style-type: none"> • Must be unique • Must start with an alphabetic character • Can contain only alphanumeric characters, underscores, and periods • Can contain up to 64 characters • Cannot include spaces • Cannot begin with a period, and cannot contain two consecutive periods <p>We recommend that you assign IDs that contain a minimum of two periods, for example, <code>com.sap.mobile.app1</code>.</p> <p>If you are building an Android application, its ID must follow the Google defined rules, or the Android build fails:</p> <ul style="list-style-type: none"> • The ID must have at least two segments (one or more periods). • Each segment must start with a letter. • All characters must be alphanumeric or an underscore [a-zA-Z0-9_].
Name	The application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.
Description	(Optional) The description can contain up to 255 alphanumeric and special characters.
Vendor	(Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long.
License Type	The license type that you want to use for your account: "resources" (default), "build-code", "free", "standard (Users)", or "b2c (Consumer Edition)". The "kernel-service" plan is used for an application that is managed in a single-app cockpit (Software as a Service, SaaS). The "lite" plan is available only for trial subaccounts. The license determines capabilities and charges. See <i>Service Plans</i> and <i>Changing Service Plans at the App Level</i> for additional information.
Domain of Application Route	<p>In regions that support multiple domains, select the application route through which end users can reach the application. Routes are associated with a space, and are configured in SAP BTP cockpit. The drop-down list shows all shared and custom domains that are available for the customer organization.</p> <div> <p>Note</p> <p>If you want to use a custom domain, you can create one according to information in What Is Custom Domain.</p> </div>

Field	Value
Client Connection Timeout	<p>(Optional) The maximum time in milliseconds before a client connection times out in your environment. After that timeout period, the connection is closed. If set to 0, the timeout is disabled. Default is 120000 milliseconds.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Session Timeout	<p>(Optional) The number of minutes a session can remain idle before the server terminates it automatically. Default value is 15 minutes.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Enable Compression	<p>(Optional) Compresses resources before sending them to the client. By default text resources above 1KB is compressed.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Zero Maintenance Downtime	<p>(Optional) Indicates whether zero maintenance downtime is enabled.</p> <p>Note No longer appears after upgrading to mobile application service.</p>

- On the [Security Settings](#) page, enter configuration values and then select [Next](#).

For environments where Identity Authentication service (IAS) has been configured, you have the option to select either the IAS (Identity Authentication service) or XSUAA (extended UAA) security configuration when you create a new mobile application. See *Configuring IAS Security* for information about configuring Identity Authentication service.

- [XSUAA Settings](#) - select this option to use the SAP Authorization and Trust Management service method for authentication.
- [IAS Settings](#) - select this option to use the SAP Identity Authentication service method for authentication.

Note

You cannot switch a mobile application between XSUAA and IAS, but you can switch between the default and existing instances.

See the following tables for configuration values, depending on your selection. You can modify these values later on the [Security](#) tab, as described in *Configuring App Security*.

XSUAA Settings

Field	Value
XSUAA Service	<p>Select the XSUAA authentication and authorization service to use. You can select Default Instance to create a new instance, or you can select an existing service from the list.</p> <p>Using an existing XSUAA instance is useful if you already have a Cloud Foundry application deployed and want to connect from Mobile Services to this application. In this case it can be handy to re-use an existing XSUAA instance.</p> <p>Select the default instance for other scenarios.</p>
xs-security.json	<p>A JSON file that defines the authentication methods and authorization types used to access your application. See Application Security Descriptor Configuration Syntax for additional information.</p>
Token Lifetime	<p>Enter the token lifetime, and select the units (Hours or Minutes). The default token lifetime is 15 minutes, and the allowed range is 10 minutes to 24 hours.</p> <p>See Application Security Descriptor Configuration Syntax for additional information.</p>
Refresh Token Lifetime	<p>Enter the refresh token lifetime, and select the units (Days, Hours, or Minutes). The value must be greater than the Token Lifetime value.</p> <p>See Application Security Descriptor Configuration Syntax for additional information.</p>
Approved Providers	<p>Indicate whether the app should support all approved providers or only selected providers.</p>
System Attributes in Token	<p>Indicate groups, role collections or both.</p>


IAS Settings

Field	Value
IAS Service	<p>Select the IAS authentication and authorization service to use. You can select Default Instance to create a new instance, or you can select an existing service from the list.</p>
Security-config.json	<p>A JSON file that defines the authentication methods and authorization types used to access your application.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>

Field	Value
Token Lifetime	<p>Enter the token lifetime, and select the units (Hours or Minutes). The default token lifetime is 15 minutes, and the allowed range is 10 minutes to one hour.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>
Refresh Token Lifetime	<p>Enter the refresh token lifetime, and select the units (Days, Hours, or Minutes). The default refresh token lifetime is 30 days, and the allowed range is 1 hour to 180 days.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>

- On the [Role Settings](#) page, enter roles and then select [Next](#).

The [Enable Role Settings](#) option is selected by default. When selected you must add one or more roles to the list. Only users who are assigned these roles can access the app, which helps ensure better security. SAP recommends enabling roles.

Enter one or more role names, separated by either commas or the [Return/Enter](#) key, for example "Developer, Sales, Manager". To learn more about default Cloud Foundry roles, see [Orgs, Spaces, Roles, and Permissions](#) .

Once the roles have been entered, you can maintain them through the [Security](#) tab for the app as described in [Configuring App Security](#).

- On the [Assign Features](#) page, select one of the configuration templates.

- Native – applications built using the native SDKs for iOS or Android.
- Mobile Development Kit - metadata-based applications.
- Hybrid – Kapsel container-based applications.

The features selected for the template appear. You can select or remove features (except for required features) at this time. Later you can make further changes as described in [Managing Application Features](#).

Basic Features

Feature	Description
Mobile App Catalog	Upload mobile application artifacts for beta testing, and for deployment to external services.
Mobile App Update	Upload new versions of a hybrid or Mobile Development Kit application.
Mobile Augmented Reality	Allows you to manage client augmented reality resources that can be accessed from mobile applications.
Mobile Settings Exchange	Handles device registrations and provides exchange of general settings between mobile client and server, such as client policies.

Feature	Description
Mobile Client Resources	Add client resources to an application.
Mobile Connectivity	Configure routes to back ends.
Mobile Offline Access	Enable secure, offline access to data on the device.
Mobile Push Notification	Register devices to receive native push notifications.
Mobile Client Log Upload	Enable application to upload application log files and to analyze them on the server.
Mobile Network Trace	Enable application to trace network activity based on user name or content type.
Mobile Sample OData ESPM	Use OData sample service during development and testing.
Mobile Client Usage and User Feedback	Enable application to upload client usage and feedback information and analyze it on the server.
Mobile Cloud Build	Build binaries for your standard and customized Mobile Development Kit (MDK) and SAP Asset Manager clients, and enable them to use the SAP Mobile Platform SDK.

6. Select *Finish* and confirm to create the application definition. The Info tab appears with current settings.

Related Information

[Managing Application Features \[page 51\]](#)

[Configuring Assigned Features \[page 53\]](#)

[Configuring App Security \[page 282\]](#)

[Service Plans \[page 23\]](#)

1.5.5.1.2 Managing Application Features

You can add features to an application from the Features menu option. Features enable capabilities that developers have added to an application.

Procedure

1. In Mobile Services cockpit, select *Features*.

Basic Features

Feature	Description
Mobile App Catalog	Upload mobile application artifacts for beta testing, and for deployment to external services.
Mobile App Update	Upload new versions of a hybrid or Mobile Development Kit application.
Mobile Augmented Reality	Allows you to manage client augmented reality resources that can be accessed from mobile applications.
Mobile Settings Exchange	Handles device registrations and provides exchange of general settings between mobile client and server, such as client policies.
Mobile Client Resources	Add client resources to an application.
Mobile Connectivity	Configure routes to back ends.
Mobile Offline Access	Enable secure, offline access to data on the device.
Mobile Push Notification	Register devices to receive native push notifications.
Mobile Client Log Upload	Enable application to upload application log files and to analyze them on the server.
Mobile Network Trace	Enable application to trace network activity based on user name or content type.
Mobile Sample OData ESPM	Use OData sample service during development and testing.
Mobile Client Usage and User Feedback	Enable application to upload client usage and feedback information and analyze it on the server.
Mobile Cloud Build	Build binaries for your standard and customized Mobile Development Kit (MDK) and SAP Asset Manager clients, and enable them to use the SAP Mobile Platform SDK.

-
2. (Optional) To find out more information about a feature, select its name.

Under [Feature Details](#), read the feature summary.

-
-
3. Click [Add to Application](#).
4. In [Select Application](#), select an application from the list.

You can filter the list of options.

Related Information

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)


1.5.5.1.3 Configuring Assigned Features

Configure features associated with an application definition.

Context

You can add additional features, and delete features that are not required. Some features require coding in the application. In general, there should be no dependency between features. For some features you can configure a service key, which enables an application to access a service instance using the service key as its credentials (see [Service Keys](#)).

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Under [Assigned Features](#), click the Add icon , select a feature, and click [OK](#).
3. Configure the feature for the selected application, and [Save](#).
4. (Optional) Select [Enable Detailed Event Log](#) and [Disable Detailed Event Log](#) to toggle event logging on and off at the feature level.
5. (Optional) Select [Reset](#) to reset settings to default values.
6. (Optional) Select [Remove from Application](#) to remove the feature from the application.

[Uploading and Deploying Apps \[page 55\]](#)

If the selected app uses the AppUpdate plugin, you can activate a new version of the app using the [Mobile App Update](#) feature.

[Configuring Mobile Client Log Upload \[page 58\]](#)

Configure mobile client log upload for the application, and for individual users.

[Defining Client Policies \[page 61\]](#)

Set client policies for the selected mobile application and manage its user registrations.

[Uploading Client Resources \[page 81\]](#)

Upload client resources (or resource bundles) and manage service keys for the selected application.

[Configuring Client Usage and User Feedback \[page 82\]](#)

Configure the [Mobile Client Usage and User Feedback](#) feature to enable the application to upload client-specific usage data reports and feedback to the server; manage files that support crash dump processing; view user groups, and manage service keys.

[Defining Connectivity \[page 89\]](#)

Define destinations for the selected application. You can also edit Mobile and On-Premise destinations.

[Tracing Network Activity \[page 116\]](#)

Trace network activity between mobile services, apps and other services, and download tracing information in a .zip file.

[Editing JSON Storage \[page 118\]](#)

Enable and manage persistent JSON storage for the selected application.

[Defining Offline Settings for Applications \[page 119\]](#)

Define offline settings for the selected application. Offline support enables client applications to access back-end data without a connection. When offline, applications access data from an offline store on the client. SAP Mobile Services moves data between the back end and the client offline store.

[Using the Mobile Sample OData ESPM \[page 136\]](#)

A sample OData service is available for developers to use during development and testing. The sample OData service also lets you evaluate how delta tokens are handled in your test application. The sample back end should not be used for production.

[Defining Push Notifications \[page 141\]](#)

Configure push-related settings for the selected application.

[Configuring Mobile Augmented Reality \[page 153\]](#)

(Native/MDK only) Use the Mobile Augmented Reality feature to manage client augmented reality resources that can be accessed from mobile applications. For example, a training or museum app may be annotated to provide access to detailed information.

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

(Not available in all COUNTRY/REGIONS) Use the cloud build service to configure and build clients in Mobile Services cockpit, and enable them to use the SAP BTP SDK.

[Managing Application Versions Using App Lab \[page 175\]](#)

Developers can manage internal application versions during the development and test cycle using the App Lab service.

[Exporting Data \[page 178\]](#)

SAP Mobile Services allows you to export business configuration data for an assigned mobile service that is bound to an application.

Related Information

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)

[Service Plans \[page 23\]](#)

1.5.5.1.3.1 Uploading and Deploying Apps

If the selected app uses the `AppUpdate` plugin, you can activate a new version of the app using the [Mobile App Update](#) feature.

Prerequisites

You need a Mobile Development Kit or hybrid (Kapsel) app package.

The hybrid app package must:


- Contain the contents of the application's `www` folder and the `config.xml` project file, with a separate folder in the archive for each mobile platform (`android/www` and/or `ios/www` in all lowercase). The format structure for a hybrid apps is:

```
| - android
|   | - config.xml
|   | - www
| - ios
```

- Be compressed into a standard `.zip` file for upload.


You can subscribe to MDK project build notices, and manage them via [My Alerts](#). If you create a condition for the subscription via the SAP BTP Alert Notification service (ANS), you can continue to receive alert notifications for three months. You can extend the subscription for another three months using [My Alerts](#).


Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile App Update](#) under [Assigned Features](#) (or add it first).
3. Select [Configuration](#) and then  to import a new application or update an existing application version.
 - a. In the [Import Application Version](#) dialog, select [Browse](#), and navigate to the directory that contains the MDK or hybrid app package.
 - b. Select the package, and select [Upload](#).

New version information appears for the uploaded MDK or hybrid app for each mobile platform. You cannot change this information.

MDK/Hybrid App Properties


Property	Description															
Status	<p>State of the MDK or hybrid app version:</p> <ul style="list-style-type: none">New Version – a newly uploaded version.Staged Version – in testing. A user who is defined as a tester can download and test applications. Once testing is complete, an administrator can promote a version to the Current state, so it becomes active. If testing fails, the administrator can change the state back to New.Current Version – the version that is currently active. <p>Moving Applications Between States</p> <table><tr><th>Beginning State</th><th>Action</th><th>Ending State</th></tr><tr><td>New</td><td>Click Stage</td><td>Staged</td></tr><tr><td>New</td><td>Click Deploy</td><td>Current</td></tr><tr><td>Staged</td><td>Click Remove</td><td>New</td></tr><tr><td>Staged</td><td>Click Deploy</td><td>Current</td></tr></table> <div><div> Note</div><p>Each platform can have an unlimited number of versions in the New state, but only one version in the Staged state, and one version in the Current state.</p></div>	Beginning State	Action	Ending State	New	Click Stage	Staged	New	Click Deploy	Current	Staged	Click Remove	New	Staged	Click Deploy	Current
Beginning State	Action	Ending State														
New	Click Stage	Staged														
New	Click Deploy	Current														
Staged	Click Remove	New														
Staged	Click Deploy	Current														
Operating System	The operating system on which the application runs, such as Android, iOS, and MDK.															
Type	The application type, such as MDK and Hybrid..															
Description	The application description, such as Mobile Development Kit App.															
Versions	<p>The app versions, depending on the app type.</p> <ul style="list-style-type: none">For MDK apps, Metadata Version is assigned to the collective settings used to build the app. The download action is supported.If no version appears, this means that the metadata version was not supported when the file was created. The download action is not supported.For Hybrid apps, Required Client Version identifies the SDK version that was used to develop the app, for example, 3.0.0. This version attribute is informational only; it is not used to determine whether device clients should receive a Web application update. Development Version identifies the internal development version that was used to develop the hybrid app.															
Uploaded By	The user who uploaded the hybrid or MDK app.															

Property	Description
Revisions	Identifies the production version revision. A revision number is assigned to a newly uploaded hybrid app and incremented when a new version is uploaded. When the hybrid app is deployed, the revision number is incremented.
<div>  Note </div> <p>The server limits the number of old active revisions that are supported. Currently the maximum number of old active revisions is set to 3. Once the limit is reached, the oldest versions are removed.</p>	
Actions	Actions to perform such as Deploy, Stage, or Delete. If an icon does not appear, the action is not supported for the version.

- To deploy applications, identify the application you want to deploy, and then select [Deploy](#).

Deployed app information appears as the current version, and the revision number is incremented.

For device-application users:

- If a hybrid app with the default version (revision = 0) connects to the server, the server downloads the full hybrid app.
 - If a hybrid app with a version (revision = 1 or higher) connects to the server, the server calculates the difference between the user's version and the new version, and downloads a patch containing only the required changes.
 - If the application implements the `AppUpdate` plugin, the server checks for updates when the application starts or resumes. If the developer has made changes, `AppUpdate` detects them using the `www` folder content (the HTML-based content), and not with native plugins or changes made outside of that folder. For changes made outside the `www` folder, the developer must post a new copy of the app to the application download site, or use app stores or use mobile device management to push the new app to all users.
- To remove application versions, identify an application and then select . When deleting extension bundles, this deletes all revisions, whether deployed or not. You can delete extension bundles for both provider and subscriber accounts.

To optimize life-cycle management for hybrid applications and to provide more efficient client updates, mobile services archive a limited number of applications that a client has previously downloaded to his or her device. If a client requests an application update and the client version of the application is available, the delta version is sent to the client; if the client version is not available, the full version is sent to the client.

- Select [Info](#) to view feature details. You can also use [Export Data](#) to download mobile service data in JSON format.

1.5.5.1.3.2 Configuring Mobile Client Log Upload

Configure mobile client log upload for the application, and for individual users.

Prerequisites

Requires the Mobile Client Log Upload service.

Context

You can specify how long to retain logs before they are deleted; view fatal and error logs, or logs for all log levels. You can view important URLs, and export mobile service data in JSON format.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ► or *SAP Mobile Cards*.
2. Select an application, then select *Mobile Client Log Upload* under *Assigned Features* (or add it first).
3. Select *Error Logs* to view error and fatal client logs for the application.

Enter search criteria to narrow the focus and then select *Go*. You can search based on logging *Level*, *Correlation ID* and *Time Frame* (either a predefined time frame, or a custom time frame using the calendar date-picker to define a range).

You can sort the search results, view details for a client log, and download the log in JSON format.

Note

The download action is only available for logs uploaded after the July 2021 release. The downloaded log file contains the specific error log and other log-level logs related to the error log. For logs uploaded before this release, the download option is not available.

Error Log Files

Column	Description
Time (UTC+0000)	The log time in UTC format.
Level	The log level, including Fatal or Error.
Message Text	The log message text.
User Name	The user that uploaded the log file containing the log.

Column	Description
Location	The location, such as <code>com.<company>.class</code> .
Action	The action to take, such as viewing the log details or downloading the log file (in JSON format) containing the log to your local environment.

4. Select [Configuration](#) and configure default and user-specific client log upload properties.
 - a. Under [Default Logging](#), configure server and client-side logging properties. The default values are used, unless user-specific values are provided.

Default Logging Properties

Property	Description
Log Upload	Whether the server is enabled to upload client logs.
Log Level	<p>The default client-side log level setting for all users.</p> <ul style="list-style-type: none"> • None – no log level is set. • Debug – for debugging purposes, includes extensive and low-level information. • Information – informational text, used mostly for echoing what has been performed. This is the default and recommended setting, unless you are actively troubleshooting or seeking specific information. • Warning – the application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator. • Error – the application can recover from the error, but cannot fulfill the task due to the error. • Fatal – the application cannot recover from the error, and the severe situation causes fatal termination. • Path – for tracing execution flow. Used, for example, in the context of entering and leaving a method, looping, and branching operations. (Not applicable to the offline logging component.)
Delete Uploaded Log After	Select the time period after which client logs are deleted from the database. By default, log files exist in the database for seven days.
Maximum Number of Logs	Enter the maximum number of log files that can be stored on the user's system at the same time. Choose a value from 1-99999.

Property	Description
Maximum Log Size	Enter the maximum size of each log file. Choose a value from 0-100 MB. '0 MB' indicates that the file size is unlimited.

Note

If the maximum number of log files or file size exceeds the defined value, the oldest log file will be deleted and the logs will be written to a new log file. This can happen irrespective of whether the logs are uploaded to the server or not. To ensure the logs are retained, you can set the maximum number of log files to "1" and file size to 0 MB.

- b. Under [User Specific Logging](#), view existing user-specific logging values. User-specific values override the default values for that user.

Select  to add a user, or edit existing values, using the guidelines provided

- a. Select [Save](#).
5. Select [Log Files](#) to view log files that have been uploaded, including all levels, not just error or fatal. A log file is created for each upload. Note that this applies to log files that are uploaded starting with this release. Log files uploaded prior to the July 2021 release will not be shown here.

You can select one or more log files, and use [User Name](#) and [Time Frame \(UTC+0000\)](#) to filter and search the logs. You can select one or more log files to download the files in JSON format to your local environment.

Log Files

Column	Description
User Name	The user that uploaded the log file.
Uploaded (UTC+0000)	The date and time the file was uploaded, in UTC format.
Lines	The total number of logs in the log file.
Line with Error	The number of logs in the log file with a log level of ERROR or FATAL.
Start Timestamp (UTC+0000)	The timestamp (in UTC) of the first log in the log file.
End Timestamp (UTC+0000)	The timestamp (in UTC) of the last log in the log file.

6. Select [Info](#) to see feature details, such as feature description and useful URLs, and to export mobile service data in JSON format.

1.5.5.1.3.3 Defining Client Policies

Set client policies for the selected mobile application and manage its user registrations.

Mobiles Services uses the Mobile Settings Exchange feature to manage user registrations and to exchange general settings, such as client policies, between mobile client and server.

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ► or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. Select [Client Configuration](#) to configure policies.
4. Select [User Registrations](#) to manage registered application users. See [Managing Registered Users](#) for information.
5. Select [Service Keys](#) to implement a service key, enabling an application to access a service instance using a service key as its credentials. The service must be able to support service keys. See [Service Keys](#) for information.
6. Select [Info](#) to view feature details, and to download mobile service data in JSON format. For applications configured for OAuth security, you can copy connection settings from one place.

[Defining Client Passcode Policy \[page 62\]](#)

Define the client passcode policy used to unlock the DataVault, for the selected application. Application developers must add code to enforce the policy to the DataVault used by the application. An administrator enters the application passcode policy used to unlock the DataVault during application initialization.

[Defining Application Management Policies \[page 63\]](#)

Define application management policies from Mobile Services cockpit for the selected app, such as whether to detect device compliance; restrict users from cutting, copying, and pasting information between apps; and restrict users from restricting printing data or opening URLs.

[Defining Device Compliance Policy \[page 64\]](#)

Define whether Mobile Services server should check for client device compliance. If enabled, Mobile Services server checks whether devices have been jailbroken or rooted.

[Defining Lock and Wipe Policy \[page 66\]](#)

Define the policy for locking and wiping the application running on a device.

[Defining Network Synchronization Policy \[page 67\]](#)

Define the policy for synchronizing application components on various channels, including WiFi, mobile networks, and roaming.

[Defining Device Policy Profiles \[page 68\]](#)

Create device policy profiles for the selected app, and the conditions that trigger the policy profile. For example, you may want to apply one passcode policy to one group of users, and another policy to another group.

[Defining Feature Restriction Policy \[page 70\]](#)

Feature restriction policies enable you to control feature access for the selected application. Set these policies using feature flags in Mobile Services cockpit. You can add, allow, restrict, edit, or delete features.

[Defining Feature Restriction Policy \(SaaS Providers\) \[page 74\]](#)

Describes the end-to-end process for software-as-a-service providers to set feature restriction policies for their tenants.

[Enabling Shared Devices \[page 80\]](#)

(Native SDK only) For environments where employees share devices, you can enable the uploading of pending changes from previous device users. This is useful in case employees forget or are unable to upload their work.

Related Information

[Revoking OAuth Tokens \[page 301\]](#)

[Managing Registered Users \[page 219\]](#)




1.5.5.1.3.3.1 Defining Client Passcode Policy

Define the client passcode policy used to unlock the DataVault, for the selected application. Application developers must add code to enforce the policy to the DataVault used by the application. An administrator enters the application passcode policy used to unlock the DataVault during application initialization.


Context

The client passcode policy applies only to the application passcode that unlocks the DataVault during application initialization; it affects neither SAP Mobile Services security profiles nor the back-end security systems with which it integrates. Passcode policies for back-end security systems are administered by customer information technology departments using native security administration tools.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. On [Client Configuration](#), under [Passcode Policy](#), select [Enable Passcode Policy](#), and enter:

Property	Default	Description
Expiration Time Frame Days	0	The number of days a passcode remains valid. The default value, 0, means the passcode never expires.
Minimum Length	8	The minimum passcode length.
Retry Limit	20	The number of retries allowed when entering an incorrect passcode. After this number of retries, the client is locked out, the DataVault and all its contents are permanently deleted, the application is unusable, and encrypted application data is inaccessible.

Property	Default	Description
Minimum Unique Characters	0	The minimum number of unique characters required in the passcode.
Lock Timeout	0	The number of seconds the secure store remains unlocked within an application, before the user must reenter his or her default passcode to continue using the application (similar to a screen-saver feature).
No Passcode Required	Disabled	If enabled, a default passcode can be generated by the DataVault. From the user's point of view, this policy turns off the passcode.
<div>  Note If the default passcode is set, the SDK automatically generates a default passcode to encrypt the DataVault. The default passcode is encrypted by an operating system-level encryption mechanism and stored securely. </div>		
Biometric Authentication Allowed	Disabled	If enabled, it allows the use of native biometric techniques to unlock the app.
Upper Case Character Required	Disabled	If enabled, the passcode must include uppercase letters.
Lower Case Character Required	Disabled	If enabled, the passcode must include lowercase letters.
Special Character Required	Disabled	If enabled, the passcode must include special characters.
Digits Required	Disabled	If enabled, the passcode must include digits.

4. Select [Save](#).

1.5.5.1.3.3.2 Defining Application Management Policies

Define application management policies from Mobile Services cockpit for the selected app, such as whether to detect device compliance; restrict users from cutting, copying, and pasting information between apps; and restrict users from restricting printing data or opening URLs.

Context

These policies can provide some additional security. By default the options are not enabled.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#) or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#)
3. On [Client Configuration](#), under [Application Management Policies](#), enable or disable the policies. Note that some options may not appear for some apps in SaaS or PaaS environments.
4. For [Device Compliance Detection](#), select or deselect the checkbox to enable or disable the restriction. When enabled, Mobile Services checks whether the device is jailbroken (iOS) or rooted (Android). The option is not enabled by default. For more information see [Defining Device Compliance Policy](#).
5. For [Restrict Cut, Copy and Paste between Apps](#), select or deselect the checkbox to enable or disable the restriction. This feature defines whether users can cut, copy, and paste information between apps in Mobile Services cockpit. When enabled, this can provide some additional security by limiting the exchange of information between apps. The option is not enabled by default.
6. For [Restrict Print Data](#), select or deselect the checkbox to enable or disable the restriction for the app. When enabled, a user is prevented from printing data from the application. This would be useful in a test scenario or when data is highly sensitive. The option is not enabled by default.
7. For [Restrict Opening URLs](#), select or deselect the checkbox to enable or disable the restriction for the app. When enabled, a user is prevented from accessing certain URLs. This would be useful when an URL can only be accessed by certain roles. The option is not enabled by default.
8. Select [Save](#).

Related Information

[Defining Device Compliance Policy \[page 64\]](#)

1.5.5.1.3.3.3 Defining Device Compliance Policy

Define whether Mobile Services server should check for client device compliance. If enabled, Mobile Services server checks whether devices have been jailbroken or rooted.

Context

Jailbreaking is a term used for iOS devices, while rooting is a term used for Android devices. Jailbreaking or rooting a device refers to the process of removing software restrictions imposed by the manufacturer or carrier on the device, allowing the user to access the device's file system and install unauthorized apps, among other things. However, jailbreaking or rooting a device can also make it more vulnerable to security threats and malware.

This feature provides some additional security by detecting compromised devices. The option is not enabled by default. Once enabled, Mobile Services server checks device compliance and reports status. Status options include:

- Compliant - the device has undergone compliance detection, and was found to be neither jailbroken (iOS) or rooted (Android).
- Compromised - the device has undergone compliance detection, and was found to be jailbroken (iOS) or rooted (Android). The compromised device status is also reported as an Error in the Event Log (use the Mobile Device Security filter). Once the compliance problem is fixed and rechecked, the status changes.
- Unknown - the device has not undergone compliance detection, so its status is not yet known.

When compromised devices are detected, they are reported as a Mobile Device Security event, and appear on the [User Registrations](#) tab and in the Event Log.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#) or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#)
3. On [Client Configuration](#), under [Compliance Policy](#), select [Enable Compliance Detection](#) to enable the restriction.
4. When the policy is enabled, Mobile Services SDK regularly checks for the latest security policy settings. If the compliance check policy is enabled, the SDK initiates the check and reports to the server if security compliance checks fail. If a compromised device is detected, a Mobile Device Security event is created.

You can:

- Check [User Registrations](#) to view the device status for an app. See *Defining Client Policies* for information.
 - Check the Event Logs for details about the Mobile Device Security event. See *Viewing Event Logs* for information.
 - Subscribe to Mobile Device Security events using information in *Subscribe to Event Log Alerts*.
5. Optionally you can turn compliance detection off if needed.

Related Information

[Defining Client Policies \[page 61\]](#)

[Viewing Event Logs \[page 237\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

1.5.5.1.3.3.4 Defining Lock and Wipe Policy

Define the policy for locking and wiping the application running on a device.

Prerequisites

(Native/MDK, SAP Build Apps, and Mobile Cards) The app must include SDK logic to support locking and wiping policies. For example, the developer may implement locking and/or wiping for when a device is lost or stolen, or a user wants to delete data stored locally.

Context

From Mobile Services cockpit, the administrator can create locking and wiping policies for the app:

- Locking refers to locking the app on the device client. Once the app is locked, the user can unlock it by connecting to the server and authenticating. All existing data (in particular the offline OData store) remains on the device. You can set the number of days before locking occurs.
- Wiping refers to resetting the application on the device. This deletes existing data on the device. You can set the number of days before wiping occurs.

When setting locking and wiping values in the cockpit, keep in mind these validation rules:




- When the lock or wipe policies are not enabled, the fields show a value of zero (0).
- When the lock or wipe policies are enabled, at least one of the fields must have a non-zero value.
- The lock period must be less than the wipe period.
- The number of days before wiping must be greater than the number of days before locking.

The server informs the app when the policy takes effect, and the application responds according to its SDK programming.

Note

The locking policy is not the same as the blocking policy, which prevents the application user from registering the application or receiving traffic. See *Managing Applications*.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Settings Exchange](#) under *Assigned Features*.
3. Select [Client Configuration](#). Under *Locking and Wiping Policy*, select [Enable Locking and Wiping Policy](#).
4. Configure the locking and wiping features, keeping in mind the validation rules previously described

Locking and Wiping Policy

Policy	Description
Offline Days Before Locking	<p>Set the number of days before the application will be locked on the user device. The number of days before locking must be lower than the number of days before wiping. Zero (0) indicates locking is not enabled.</p> <p>If locked, the user must reconnect to the server and authenticate.</p>
Offline Days Before Wiping	<p>Set the number of days before the application is reset on the user device. The number of days before wiping must be higher than the number of days before locking. Zero (0) indicates locking is not enabled.</p> <p>If wiped, the application remains, but the data is deleted.</p>

5. Select [Save](#).

1.5.5.1.3.3.5 Defining Network Synchronization Policy

Define the policy for synchronizing application components on various channels, including WiFi, mobile networks, and roaming.

Prerequisites

The developer must include a user interface on the client app for users to configure when to synchronize large data uploads.

Note

This feature is available only for apps created with SAP SDK for iOS and SAP SDK for Android.

Context

This feature lets administrators control synchronization behavior for each app from the cockpit, by indicating when it is okay for data-heavy SDK components to synchronize. Eligible components include:

- Analytics
- Client resources
- Logs
- Offline OData

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK** or **SAP Mobile Cards**.
2. Select an application, then select **Mobile Settings Exchange** under **Assigned Features**.
3. Under **Network Synchronization Policy**, select **Enable Network Policy**.
4. For each component in the list, select each network channel to use for synchronizing data.

Network Synchronization Policy

Policy	Description
Component	The component that's impacted by the network synchronization policy (such as Analytics, Client Resources, Logs, and Offline OData).
WiFi	Indicates whether synchronization should automatically occur over WiFi.
Mobile Networks	Indicates whether synchronization should automatically occur over mobile networks. Network provider charges may apply.
Roaming	(Android only) Indicates whether synchronization should automatically occur while roaming. Network provider charges may apply.

5. Select **Save**.

1.5.5.1.3.3.6 Defining Device Policy Profiles

Create device policy profiles for the selected app, and the conditions that trigger the policy profile. For example, you may want to apply one passcode policy to one group of users, and another policy to another group.

Prerequisites

This feature is available:

- Version 2 API only
- Native/MDK apps only
- Includes SaaS apps

Context

This feature provides a way to divide users into different groups in Mobile Services cockpit, and use conditions to apply different policies to each group. These policies can be applied to device profiles:

- Pass code Policy
- Locking and Wiping Policy
- Network Synchronization Policy

Once the policies are created, you can use drag and drop to adjust the order of the profiles. The profile at the top of the list is applied first followed by each subsequent profile.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#) or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. Select [Client Configuration](#). Under [Device Policy Profile](#), you can view the list of device policy profiles..

Device Policy Profiles

Property	Description
Name	Descriptive name used to identify the device policy profile.
Description	A more detailed description of the device policy profile.
Active	Whether the policy is active.
Actions	Actions you can take such as edit or delete.

Note

If you change the [Active](#) setting, be sure to select [Save](#) in the upper right corner of the page to save the change.

4. Select [+](#) to add a new profile using the [New Device Policy Profile](#) wizard, or select an existing profile to edit it.
 - a. On the [General Information](#) page, enter basic information and then select [Next](#).

General Information

Property	Description
Name	Descriptive name to identify the device policy.
Description	(Optional) Description of the device policy with additional details.
Active	Indicates whether the policy is currently active.

- b. On the [Passcode Policy](#) page, select [Enable Passcode Policy](#) and enter policy values. See [Defining Client Password Policy](#) for details. When finished, select [Next](#).

- c. On the [Locking and Wiping Policy](#) page, select [Enable Locking and Wiping Policy](#) and enter policy values. See [Defining Lock and Wipe Policy](#) for details. When finished, select [Next](#).
- d. On the [Network Synchronization Policy](#) page, select [Enable Network Policy](#) and enter policy values. See [Defining Network Synchronization Policy](#) for details. When finished, select [Next](#).
- e. On the [Triggered Conditions](#) page, enter the conditions for which the device policy profile is activated, in one or more condition groups.
 1. If necessary, select [Add a Condition Group](#) to get started.
 2. Add one or more conditions to the group.

Condition Properties

Property	Description
SAML Attribute Name	The attribute name for the condition.
Operator	The operator to use for the condition, such as Equals or Contains.
SAML Attribute Value	The value to use.
Actions	The action to take, such as delete a condition or group, or add another condition to the condition group.

3. Add one or more additional groups as needed.
5. Select [Finish](#) to save. The wizard closes, and the policy appears in the device policy profiles list.
6. Once the policies are created, you can use drag and drop to adjust the order of the profiles. Select [Save](#) in the upper right corner of the page to save the changes.

The profiles are applied in the order they appear, from the first in the list to the last.

1.5.5.1.3.3.7 Defining Feature Restriction Policy

Feature restriction policies enable you to control feature access for the selected application. Set these policies using feature flags in Mobile Services cockpit. You can add, allow, restrict, edit, or delete features.

Context

When you edit a native or hybrid app, or an app developed with the Mobile Development Kit, in Mobile Services cockpit, available feature plugins are listed on the [Mobile Settings Exchange](#) screen. Feature plugins are typically JavaScript APIs that provide access to the native APIs of the mobile device (for example, for Hybrid apps they are typically implemented as Apache Cordova plugins, such as Camera, Calendar, and Push). You can indicate features that should be restricted from the user.

When a plugin, for example, the barcode scanner plugin, is in a disabled state on the server, the application starts a settings exchange and does two things:

- Invalidates the native side of the plugin

- Changes the namespace of the plugin to `null`

You may later enable a plugin on the server side, and trigger a settings exchange. At that time, although the plugin is not present in the disabled list, the value of the `cordova.plugins.barcodeScanner` namespace remains `null`; this value is reset only if a page refresh occurs, and Cordova reloads the plugin namespaces.

The new feature restriction policy takes effect after you exit the application and restart it to allow Cordova to refresh all the namespaces.

Note

Currently, the SAP Fiori Client forcefully disables features that are explicitly disabled at the server through settings exchange. But it is a good idea to verify a specific feature is enabled before calling the feature in the Web application (or underlying component that consumes the application).

You can apply the feature restriction values to all users (default); or you can set feature restrictions on certain parameters to enable or disable capabilities for specific users, groups, types, and operations, and the order in which to apply them. You can specify Random Pool (A/B Testing) and choose the active / inactive percentage value.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** **Native/MDK** or **SAP Mobile Cards**.
2. Select an application, then select **Mobile Settings Exchange** under **Assigned Features**.
3. Under **Feature Flags**, you see the current status of device feature plugins.

Column	Description
Name	A unique feature plugin name.
Plugin	A list of feature plugins that are available with the application, such as Camera, Calendar, and Push.
ID	Unique identifier for the feature plugin.
Active	Indicates whether the feature is allowed or restricted. <ul style="list-style-type: none"> • On indicates the feature is allowed. • Off indicates the feature is not allowed. For some features inherited from a provider, you may be able to activate a feature, or may be restricted.
Actions	The actions you can take, such as edit or delete. For features inherited from a provider, you may be restricted from these actions.

4. (Optional) Select **+** to add a feature plugin for the selected application, or select a row to edit it, and click **OK**.

Field	Description
Name	A unique feature name.
Plugin	A list of feature plugin that are available with the application, such as Camera, Calendar, and Push.
Plugin Name	Plugin name.
Description	A feature plugin description, such as Cordova Camera Plugin, Cordova Contacts Plugin, and SAP Push Plugin.
JavaScript Module	A comma-separated list of all JavaScript modules that are used by this plugin. The JavaScript Module value is the JavaScript API that is used to invoke the plugin.
ID	Unique identifier for the plugin. The value comes from the <code>cordova_plugins.js</code> file, which appears in the project after you add a plugin ("pluginId").
Active	Toggle the feature On (allowed) or Off (restricted). By default, features are allowed.
Rule-Based Activation	Indicates whether the feature is subject to any rules. Options include: <ul style="list-style-type: none"> None - no special rules apply. The feature restriction setting applies to all app users. Activation Rule - special rules apply. You can set up one or more application rules so that the rules only apply to certain app users, and set up the order in which the rules apply. When you select this option, the Activation Rules section appears. Enter the rules in the appropriate order (described below). Random Pool (A/B Testing) - use for delayed roll out of new features to a smaller pool of random users, or for performing A/B testing on new back ends. The Active / Inactive Percentage property appears.
Active Percentage Only appears if you selected Random Pool (A/B Testing) and Active is set to Off	If you selected the Random Pool (A/B Testing) activation rule and Active is set to Off (meaning features are restricted), use the property to enter the percentage of registered app users to which you want to roll out the feature. For example, enter 25 to roll out the feature to 25% of the users.
Inactive Percentage Only appears if you selected Random Pool (A/B Testing) and Active is set to On	If you selected the Random Pool (A/B Testing) activation rule and Active is set to On (meaning features are available to all users), enter the percentage of registered app users to which you want to roll out the feature. For example, enter 35 to roll out the feature to 35% of the users.

If you selected the [Activation Rule](#), the [Activation Rules](#) section appears. Add rules in the order they should be performed.

1. Select [+](#) to add a new rule.
2. Assign the property values for the rule. See the [Activation Rule Examples](#) table for sample values.

Activation Rule Properties

Property	Description
Name	Descriptive name for the activation rule, such as "allowedPolicies: OS is Android" or "allowedPolicies: Svc Group".
Type	<p>Indicates from where to fetch information, such as from a header, a service, and so forth. The value is parsed from this source. Options include:</p> <ul style="list-style-type: none"> • App Version • OS Version • User Group • Device Platform
Operation	<p>Used to compare against the configured value.</p> <p>For App Version and OS Version, options include:</p> <ul style="list-style-type: none"> • Equals • Not Equal • Starts With • Ends With • Contains <p>Device Platform</p> <ul style="list-style-type: none"> • Equals • Not Equal • Any <p>For User Groups, options include:</p> <ul style="list-style-type: none"> • All • Any
Value	<p>The value against which the operator compares, such as these typical values:</p> <ul style="list-style-type: none"> • App Version - 1.0.0 (single value) • OS Version - iOS/16.11 (single value) • User Group - Cards_Admin, kibana_user_global (multiple values supported) • Device Platform - iOS, Android, Windows
Match	Indicates the on/off status.
Actions	The action to take, such as ✕ to remove the rule.

Activation Rule Examples

Type	Operator	Typical Value
App Version	<ul style="list-style-type: none"> • Equals • Not Equal • Starts With • Ends With • Contains 	1.0.0 (single value)
OS Version	<ul style="list-style-type: none"> • Equals • Not Equal • Starts With • Ends With • Contains 	iOS/16.11 (single value)
Device Platform	<ul style="list-style-type: none"> • Equals • Not Equal • Any 	Android (single value)
User Group	<ul style="list-style-type: none"> • All • Any 	Cards_Admin, kibana_user_global (multiple values supported)

3. Position the rules in the order they should be processed, from the first (highest in the list) to the last (lowest in the list). This results in a "match first" approach. To position the rules, select a rule and then ^ to move the rule higher up in the list, and v to move the rule lower in the list.
4. Select [OK](#) to save. The rule is added to the list.
5. (Providers only) Under [Feature Flags for SaaS Tenants](#), you see the current status of device feature plugins for your tenants, or select [Enable Feature Flags for SaaS Tenants](#) to enable the feature. See [Defining Feature Restriction Policy \(SaaS Providers\)](#) for details.

If you are not a tenant provider, this section does not appear.

6. Select [Save](#).

1.5.5.1.3.3.8 Defining Feature Restriction Policy (SaaS Providers)

Describes the end-to-end process for software-as-a-service providers to set feature restriction policies for their tenants.

Prerequisites

- The SaaS provider production landscape must be configured in SAP BTP.




- The Mobile Services cockpit must implement storage service.
- The runtime application settings must contain a `saasFeatureVectorPolicies` node.
- To activate or deactivate specific tenants, you need their tenant IDs. You can obtain the tenant IDs in SAP BTP cockpit, from the list of subaccount tenants.

Context

This feature makes it possible for software providers to make mobile application feature plugins available to tenant subscribers. The plugins can be activated for all tenants, deactivated for all tenants, or activated/deactivated for a restricted list of tenants. Depending on the settings made, the feature plugin details may be read only, or may be edited or overridden.

You can apply the feature restriction values to all users (default); or you can set feature restrictions on certain parameters to enable or disable capabilities for specific users, groups, types, and operations, and the order in which to apply them. You can specify Random Pool (A/B Testing) and choose the active / inactive percentage value.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. Under [Feature Flags for SaaS Tenants](#), select [Enable Feature Flags for SaaS Tenants](#).
4. When [Enable Feature Flags for SaaS Tenants](#) is enabled, you see the current status of device feature plugins for tenants.

Column	Description
Name	A unique feature plugin name.
Plugin	A list of feature plugins that are available with the application, such as Camera, Calendar, and Push.
ID	Unique identifier for the feature plugin.
Active	Indicates whether the feature is allowed or restricted for tenants. <ul style="list-style-type: none"> • On indicates the feature is allowed for tenants. • Off indicates the feature is not allowed for tenants.
Tenant Overridable	Indicates whether tenants can override this setting. For example, a tenant may decide they do not want their users to use a feature even though it is available.
Actions	The actions you can take, such as edit or delete.

5. Select  to add a feature plugin for tenants, or select a row to edit it and click [OK](#) to save.

Feature Flags for SaaS Tenants Properties

Property	Description
Name	Enter the feature name, such as Feature1 or SaaSFeature2.
Plugin	Enter the feature plugin.
Plugin Name	Enter the feature plugin name.
Description	Enter a description of the plugin.
JavaScript Module	Enter a list of features included in the plugin, such as Camera,navigator.camera,CameraPopoverOptions,CameraPopoverHandle.
ID	The plugin identifier, such as cordova-plugin.Feature1.
Tenant Overridable	Whether subscribers can override this setting. For example, a tenant may want to decide whether to make a feature available to its subscribers, or when to make it available.
Active	<p>Whether the feature flag is active for both the provider app (tenant) and the subscriber app. This value applies to all tenants, unless you specify otherwise.</p> <ul style="list-style-type: none"> Select On if the feature flag is active. When set to On, the feature is available to all tenants. The property "Deactive on Tenants" appears, which enables you to turn off the feature flag for specific tenants. Select Off if the feature is inactive. When set to Off, the feature is not available to any tenants. The property "Active on Tenants" appears, which enables you to turn on the feature flag for specific tenants. <p>This setting determines the default value for subscribers that appear in the Feature Flag table. See <i>Defining Feature Restriction Policy</i>.</p>
Active on Tenants	If Active is set to Off (meaning the feature is inactive for all tenants), you can use this property to override the flag for specific tenants. Enter one or more tenant IDs. The feature is then activated for the tenants in this list.
Deactive on Tenants	If Active is set to On (meaning the feature is active for all tenants), you can use this property to override the flag for specific tenants. Enter one or more tenant IDs. The feature is then deactivated for the tenants in this list.

Property	Description
Rule Based Activation	<p>Indicates whether the feature is subject to any rules. Options include:</p> <ul style="list-style-type: none"> None - no special rules apply. The feature restriction setting applies to all app users. Activation Rule - special rules apply. You can set up one or more application rules so that the rules only apply to certain app users, and set up the order in which the rules apply. When you select this option, the Activation Rules section appears. Enter the rules in the appropriate order (described below). Random Pool (A/B Testing) - use for delayed roll out of new features to a smaller pool of random users, or for performing A/B testing on new back ends. The Active / Inactive Percentage property appears.
<p>Active Percentage</p> <p>Only appears if you selected Random Pool (A/B Testing), and Active is set to Off</p>	<p>If you selected the Random Pool (A/B Testing) activation rule and Active is set to Off (meaning the feature is only available to the tenants you choose), use this property to enter the percentage of registered app users in those tenants, to which you want to roll out the feature. For example, enter 25 to roll out the feature to 25% of the users.</p>
<p>Inactive Percentage</p> <p>Only appears if you selected Random Pool (A/B Testing), and Active is set to On</p>	<p>If you selected the Random Pool (A/B Testing) activation rule and Active is set to On (meaning the feature is available to all tenants), use this property to enter the percentage of registered app users to which you want to roll out the feature. For example, enter 35 to roll out the feature to 35% of the users.</p>

If you selected the [Activation Rule](#), the [Activation Rules](#) section appears. Add rules in the order they should be performed.

1. Select [+](#) to add a new rule.
2. Assign the property values for the rule. See the [Activation Rule Examples](#) table for sample values.

Activation Rule Properties

Property	Description
Name	Descriptive name for the activation rule, such as "allowedPolicies: OS is Android" or "allowedPolicies: Svc Group".

Property	Description
Type	<p>Indicates from where to fetch information, such as from a header, a service, and so forth. The value is parsed from this source. Options include:</p> <ul style="list-style-type: none"> • App Version • OS Version • User Group • Device Platform
Operation	<p>The operator used to compare against the configured value.</p> <p>For App Version and OS Version, options include:</p> <ul style="list-style-type: none"> • Equals • Not Equal • Starts With • Ends With • Contains <p>For Device Platform, options include:</p> <ul style="list-style-type: none"> • Equals • Not Equal • Any <p>For User Groups, options include:</p> <ul style="list-style-type: none"> • All • Any
Value	<p>The value against which the operator compares, such as these typical values:</p> <ul style="list-style-type: none"> • App Version - 1.0.0 (single value) • OS Version - iOS/16.11 (single value) • User Group - Cards_Admin, kibana_user_global (multiple values supported) • Device Platform - iOS, Android, Windows
Match	Indicates the on/off status.
Actions	The action to take, such as ✕ to remove the rule.

Activation Rule Examples

Type	Operator	Typical Value
App Version	<ul style="list-style-type: none">• Equals• Not Equal• Starts With• Ends With• Contains	1.0.0 (single value)
OS Version	<ul style="list-style-type: none">• Equals• Not Equal• Starts With• Ends With• Contains	iOS/16.11 (single value)
Device Platform	<ul style="list-style-type: none">• Equals• Not Equal• Any	Android (single value)
User Group	<ul style="list-style-type: none">• All• Any	Cards_Admin, kibana_user_global (multiple values supported)

3. Position the rules in the order they should be processed, from the first (highest in the list) to the last (lowest in the list). This results in a "match first" approach. To position the rules, select ^ to move a rule higher up in the list, and v to move the rule lower in the list.
4. Select **OK** to save. The rule is added to the list.

Next Steps

Once you activate a feature for an app, it appears in the subscriber list of feature plugins. Depending on its settings, the feature may be read only, with no actions available to subscribers.

If you make the feature tenant overridable, the tenant can decide whether to make the feature plugin available to subscribers. Depending on its settings, the feature may be read only, with no actions available to subscribers.

Related Information

[Defining Feature Restriction Policy \[page 70\]](#)

1.5.5.1.3.3.9 Enabling Shared Devices

(Native SDK only) For environments where employees share devices, you can enable the uploading of pending changes from previous device users. This is useful in case employees forget or are unable to upload their work.

Prerequisites

Default and custom trust configuration must already be established in SAP BTP cockpit.

For information about establishing trust, see *Configuring Security Trust* and [Establish Trust and Federation with UAA Using Any SAML Identity Provider](#).

Context

This feature is intended for environments where devices are shared, such as when employees share devices over multiple work shifts. The upload can be executed in the back end, and the next employee can continue work the next day. The feature ensures security, so that users only receive their own messages, not messages intended for a previous user. The feature must be enabled for the application.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, then select [Mobile Settings Exchange](#) under *Assigned Features*.
3. Under *Shared Devices*, select [Allow Upload of Pending Changes from Previous User](#).

When you enable this feature, you also enable multiple user mode for the client via the QR code.

Note

In order for the Device Policy Profiles to take effect when "Multiple User Mode" is enabled, you must ensure that the Passcode Policy is also enabled as it is a part of Device Policy Profiles. See *Defining Device Policy Profiles* for information.

4. Select [Save](#).

Related Information

[Configuring Security Trust \[page 294\]](#)

1.5.5.1.3.4 Uploading Client Resources

Upload client resources (or resource bundles) and manage service keys for the selected application.

Context


Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Mobile Services. The administrator can modify the client resource bundle settings in Mobile Services cockpit. Keep in mind these resource bundle guidelines:



- Supportability – the resource bundle can be of any type (.pdf, .xls, .xml, or any other extension), with no restrictions, except that password protected zip files are not supported.
- Size – the resource bundle is restricted in size. The maximum size is 1GB, because Cloud Foundry saves the bundle content in ObjectStore DB. The maximum total bundle count is 100 for an application. For larger files, work with an application developer for performance issues.
- Default resource bundle – the first resource bundle that you upload is considered to be the default. After that, you can upload additional versions of the bundle, but only one can be the default. You can delete obsolete resource bundle versions.
- URL for the default resource bundle – `https://<mobile services host>/bundles/<ApplicationId>/`
Also supports URL – `https://<mobile services host>/mobileservices/application/{<applicationId>}/bundles/v1/runtime/bundle/application/{<applicationId>}`
- URL to access other resource bundles – `https://<mobile services host>/bundles/<ApplicationId>/<BundleName>:<BundleVersion>`
Also supports URL – `https://<mobile services host>/mobileservices/application/{<applicationId>}/bundles/v1/runtime/bundle/application/{<applicationId>}/bundle/{<bundleName>}/version/{<bundleVersion>}`

Note

To find your `<mobile services host>` name, open the application *APIs* tab. The Server URL value is also the `<mobile services host>` name.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, then select **Mobile Client Resources** under **Assigned Features** (or add it first).
3. Select **Configuration** to add client resource.
 - a. Click the **Upload Client Resource** icon  to create a new client resource:
 - **Bundle Name** – provide a name to identify the resource.
 - **Version** – provide a version number.

- [Upload Client Resource](#) – click [Browse](#), select the file, and confirm.
 - a. To define a client resource bundle as the default, select it, and click [Save](#).
 - a. (Optional) Choose  or  as required.
- 4. Select [Service Keys](#) to see service keys configured for the application feature. See [Service Keys](#).
- 5. Select [Info](#) to view feature details, and to download mobile service data in JSON format.



1.5.5.1.3.5 Configuring Client Usage and User Feedback

Configure the [Mobile Client Usage and User Feedback](#) feature to enable the application to upload client-specific usage data reports and feedback to the server; manage files that support crash dump processing; view user groups, and manage service keys.

Prerequisites

The developer must include the [Mobile Client Usage and User Feedback](#) feature API in the application.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#) .
2. Select an application, then select [Mobile Client Usage and User Feedback](#) under [Assigned Features](#) (or add it first).
3. Select [Configuration](#) to configure client usage and feedback.
4. Under [Client Usage Configuration](#), select [Enable Client Usage](#) to enable the application to collect the client's usage information.

Note

User feedback can be uploaded whether [Enable Client Usage](#) is selected.

Client Usage Settings

Settings	Description
Upload Report After	Enter the number of days after which the usage report may be uploaded to the server.
Enable Anonymization	Select to anonymize both device IDs and user names with a hash algorithm, rather than store them in plain text. This helps ensure data protection and user privacy. Toggle to disable. This feature is available only for Native/MDK apps.

Settings	Description
Enable Crash Logs Upload	Select to enable the SDK to upload crash log files from the client back end for iOS and Android apps. See <i>Managing Symbol Files (iOS)</i> and <i>Managing Mapping Files (Android)</i> to manage upload, download, and removal of files.

- Under *Client Usage Export Configuration*, select *Enable Client Usage Export Configuration* to configure parameters for exporting client usage data to an endpoint. This enables you to use data in other ways, beyond the provided Client Usage and Client Feedback reports. See *Downloading or Exporting Client Usage Data* for details about downloading usage and user feedback charts, and exporting usage and feedback data in JSON format.
- Under *Client Usage Report*, choose the number of days to include in the client usage report download.
- Under *Client Feedback Report*, choose the number of days to include in the client feedback report download.
- Select *Save* if you made changes.
- Select *Symbol Center* to manage the symbol files that can be used to debug crashes, and search for missing dSYM files for iOS. See *Managing Symbol Files (iOS)*.
- Select *Mapping File* to manage mapping files for debugging crash dumps for Android. See *Managing Mapping Files (Android)*.
- Select *User Group* to view user groups that are configured for the space. To modify the groups, see *Configuring Groups for Client Usage*.
- Select *Service Keys* to see service keys configured for the application feature. See *Service Keys*.
- Select *Info* to see feature details, such as useful URLs, and to export usage and feedback data.

[Downloading or Exporting Client Usage and User Feedback Reports \[page 84\]](#)

Download or export client usage and user feedback reports, and export data in JSON format.

[Downloading or Exporting Client Usage Data \[page 84\]](#)

Download or export client usage data to an endpoint. This enables you to securely access the data for other uses.

[Managing Symbol Files \(iOS\) \[page 86\]](#)

Manage the symbol files that can be used to debug crashes, and search for missing dSYM files for iOS.

[Managing Mapping Files \(Android\) \[page 87\]](#)

Manage mapping files for debugging crash dumps for Android.

[Configuring Groups for Client Usage \[page 88\]](#)

Configure user groups for client usage reporting. When configured you can get usage information at the user group level using Mobile Services cockpit.

Related Information

[Viewing User Data Charts \[page 248\]](#)

[Viewing User Feedback Charts \[page 249\]](#)

1.5.5.1.3.5.1 Downloading or Exporting Client Usage and User Feedback Reports

Download or export client usage and user feedback reports, and export data in JSON format.

Prerequisites

The developer must include the *Mobile Client Usage and User Feedback* feature API in the application. The app must be in active use, with data collected and uploaded to the server.

Context

You can download the client usage and the user feedback reports in Excel spreadsheet format, and you can export the data for both in JSON format.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ►
2. Select an application, then select *Mobile Client Usage and User Feedback* under *Assigned Features* (or add it first).
3. Select *Configuration*, and for *Client Usage Report*, select ⬇ to download the client usage report in CSV format.
4. For *Client Feedback Report*, select ⬇ to download the user feedback report in CSV format.
5. Select *Info*, then *Export Data* to export both the usage and feedback data in JSON format.

1.5.5.1.3.5.2 Downloading or Exporting Client Usage Data

Download or export client usage data to an endpoint. This enables you to securely access the data for other uses.

Prerequisites

The developer must include the *Mobile Client Usage and User Feedback* feature API in the application. The app must be in active use, with data collected and uploaded to the server.




Context

Configure the data upload parameters, including endpoint URL of the OData service, authorization key if needed, headers with key-value entries for authentication, and batch size.



You can download a local copy of client usage metadata in .XML format, and build an OData service based on this metadata. The OData service endpoint can be used as the URL in [Client Usage Export Configuration](#).

Data is automatically exported to the given URL one server at a time, and new records are exported hourly in batches (if the URL is left blank, data is not exported). Users can also trigger an export of new records from Mobile Services cockpit. Errors are reported to the event log (one entry per service instance per hour).

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, then select [Mobile Client Usage and User Feedback](#) under [Assigned Features](#) (or add it first).
3. Under [Client Usage Export Configuration](#), configure the export parameters. Select [Test](#) to verify the URL connection.

Client Usage Export Parameters

Parameter	Description
URL	The endpoint of the OData service. Data will be exported to this URL. If you do not want to export data to the given URL, leave the URL empty.
Authorization	The authorization key if required.
Max Batch Size	Large data sets may be exported in batches for efficiency. Indicate the number of records to include per batch. The default is 1000. Experiment to find the most efficient value to handle your data count per day. For example, for 20,000 data records per day, you determine batches of 200 records works best.
Headers	Key-value entries that can be sent as authentication to the remote service. You can enter up to ten key-value pairs. Select  to add a new key-value entry up to the limit. Select  to remove a key-value entry.

4. Select [Save](#) if you made changes.
5. (Optional) Select [Metadata Download](#) to download a local copy of client usage metadata in .XML format.

You can build your OData service based on this metadata. The OData service endpoint can be used as URL in [Client Usage Export Configuration](#).

6. (Optional) Select [Export Data](#) to manually export new client usage records outside of the automated process.

1.5.5.1.3.5.3 Managing Symbol Files (iOS)

Manage the symbol files that can be used to debug crashes, and search for missing dSYM files for iOS.

Prerequisites

The [Enable Crash Logs Upload](#) option must be enabled on the [Configuration](#) tab.

Context

Use the symbol files to troubleshoot and debug crash dumps for iOS mobile apps.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►
2. Select an application, then select [Mobile Client Usage and User Feedback](#) under [Assigned Features](#) (or add it first).
3. Select the [Symbol Center](#) tab.
4. Under [dSYMs](#) you can view the uploaded symbols files for iOS apps. To upload another symbols file, select the upload icon and browse to locate the file, and then [Upload](#). The file is added to the list. If the upload fails, it means that the uploaded file is not a valid dSYM file.

Sort by column to find specific information.

Column	Description
UUID	The universally unique identifier for the file.
File Name	The dSYM .ZIP file name. For example, <code>WeatherTracker.app.dSYM.zip</code> .
Upload Date (UTC)	The date and time the file was uploaded in UTC format, YYYY-MM-DD HH:MM:SS.
Action	The actions you can perform such as download or delete the file.

5. Under [Missing dSYMs](#) you can view all missing symbols files. The missing symbols files are required to parse the uploaded iOS crash report. To fully symbolicate a crash report, you must upload all missing dSYMs files.

You can use filters to search by UUID, Application Version, Application Name, and Image Name, and select [Go](#). You can also sort by column.

Column	Description
UUID	The universally unique identifier for the file.
Application Version	The version of the crashed application.
Application Name	The name of the crashed application.
Image Name	The name of the image.
Number of Crashes	The number of crashes reported.

1.5.5.1.3.5.4 Managing Mapping Files (Android)

Manage mapping files for debugging crash dumps for Android.

Prerequisites

The [Enable Crash Logs Upload](#) option must be enabled on the [Configuration](#) tab.

Context

Use the mapping files to troubleshoot and debug crash dumps for Android mobile apps.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, then select [Mobile Client Usage and User Feedback](#) under [Assigned Features](#) (or add it first).
3. Select the [Mapping File](#) tab.

To upload another mapping file, select the upload icon, browse to locate the file and specify the version name, and then [Upload](#). The file is added to the list.

Column	Description
File Name	The mapping .TXT file name. For example, <code>WeatherTracker.app.txt</code> .
Version Name	The version of the file, such as 2.0.
Upload Date (UTC)	The date and time the file was uploaded in UTC format (YYYY-MM-DD HH:MM:SS).

Column	Description
Action	The actions you can perform such as download or delete the file.

1.5.5.1.3.5.5 Configuring Groups for Client Usage

Configure user groups for client usage reporting. When configured you can get usage information at the user group level using Mobile Services cockpit.

Prerequisites

Groups must be configured via SAP BTP cockpit at the space level.

Context

Groups are not available for the single app cockpit in SAAS or PAAS environments. Mobile Services is limited to five user groups for the space.

Procedure

1. In Mobile Services cockpit, select  [Settings](#) > [Client Usage](#) .

Under [User Group](#) up to five user groups may appear.

2. Select [Edit](#) to make changes.

User Group Properties

Properties	Descriptions
Group ID	Up to five groups can be set at the space level. The names are set and cannot be changed, for example "Group 1" and "Group 5".
Group Name	You can modify the group name to make it more meaningful. Enter a valid name up to 256 characters long, without spaces. For example, "Organization", "Region", and "Department".

Properties	Descriptions
(Optional) From SAML Attributes	A SAML attribute such as "org", "office", and "dept". These values are from the SAML identity provider (IDP) configuration. Note that if you set a SAML attribute, the value of the SAML Token attribute binds to the value of the group ID.

3. Select [Save](#) to save your changes.

Next Steps

Once configured, when you view User Data charts you can select the [Filter](#) icon to set filters for groups, and can select [Show User Group Mapping](#) to view configuration settings for groups. See *Viewing User Data Charts* for information.

You can also see user group configuration settings at the application level, but you cannot make changes. See *Configuring Client Usage and User Feedback* for information.

1.5.5.1.3.6 Defining Connectivity

Define destinations for the selected application. You can also edit Mobile and On-Premise destinations.

Context

The Mobile Connectivity feature of SAP Mobile Services allows you to define the connectivity to back-end systems that the application can use. You can define any number of destinations to different back-ends. Those destinations are to be used exclusively by the application for which they are configured. You can restrict access to allowed paths. For applications that access Web services containing relative URLs, you can add the relative paths to enable the product to handle requests correctly. You can implement service keys for authentication.

In Mobile Services cockpit, you can view the properties of Fiori applications and connections that were developed using other tools and imported into SAP Mobile Services, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#) or [SAP Mobile Cards](#).

If you select Native/MDK app, you will view a list of mobile applications, with columns for Application ID, Name, Vendor, License Type, State, Outdated, and Creation Date.

If you select Mobile Cards, you will view a list of mobile card templates, with columns Name, Destination/SAP Client/Site ID, Status, Version, Card Template Class, Card Template Type, and Actions.

2. For Native/MDK apps, select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first).

For Mobile Cards, select [Features](#), then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first).

3. For [Configuration](#) under [Mobile Destinations](#), you can view current destinations for the selected application.

Destinations

Field	Value
Name	The destination name.
URL	The destination URL.
Rewrite Mode	For application back-end connections, the rewrite mode defines how the mobile services handles request and response messages. To enable applications that use external back ends to run offline, select one of the supported rewrite modes.
SSO Mechanism/Authentication	The single sign-on or authentication security methods employed for the destination.
Actions	<p>The actions available, such as edit or delete a connection, ping a back-end connection, and test an OData application destination.</p> <p>If an action is not supported, the icon is grayed out or absent. For example, pinging and testing OData destinations are not supported for some SSO methods. Use the Launch in Browser icon to test connectivity using the mobile application URL in a separate web browser.</p>

4. (Optional) Select [+](#), and use the [Create Destination](#) dialog to create a new destination. See [Creating a Destination](#) for details.
5. Select a row to view its settings in the Destination Overview.

The overview varies by configuration, but common sections include:

- Info – basic configuration settings.
- Rewrite Method – rewrite URL settings.
- Security – important security settings.
- Custom Headers – key:value pairs defined for static headers, or a cookie value with a variable.

Note

You can select [Edit](#) to edit some settings, or [Ping](#) to test the connection if Ping is supported for the SSO method used.

6. (Optional) From the [Service Keys](#) tab, for some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials. The feature must be able to support service keys. If you do not see the [Service Keys](#) tab for a feature, the option is not available. See [Service Keys](#).

7. (Optional) Select the *Info* tab to see useful URLs.

[Creating a Destination \[page 92\]](#)

Define a new destination to a data source or service.

[Creating a Destination with Existing Service Instances \[page 108\]](#)

Define a new destination to a back-end system using existing Cloud Foundry service instances.

[Editing a Destination \[page 109\]](#)

Modify settings for an existing destination.

[Pinging a Destination \[page 110\]](#)

Test whether a destination is accessible.

[Deleting a Destination \[page 110\]](#)

You can delete a destination only if it is not mapped to an application.

[Configuring Application-to-Application SSO Authentication \[page 111\]](#)

The application-to-application single sign-on (SSO) authentication type allows SAP Mobile Services to propagate the identity of the logged-in user to another application, which is consumed (deployed or subscribed) in the same SAP BTP account.

[Testing OData Destination Quality \[page 112\]](#)

OData destination quality testing ensures data accuracy and reliability. It is essential for ensuring that OData destinations are functioning properly and delivering high-quality data for use in various applications and systems.

[Testing an OData Destination \[page 113\]](#)

You can test destination links for OData applications from the Connections window.

[Testing Destinations in a Browser \[page 114\]](#)

You can test destination connectivity for all authentication types and for OData applications, from a web browser.

[Accessing Cloud Destinations \[page 115\]](#)

You can configure Cloud Foundry applications to use cloud destinations that are defined in SAP BTP for the customer subaccount.

Related Information

[HTTP Headers Used to Propagate User IDs \[page 396\]](#)

[Creating a Destination \[page 92\]](#)

[Accessing Cloud Destinations \[page 115\]](#)

[Service Keys \[page 303\]](#)

1.5.5.1.3.6.1 Creating a Destination

Define a new destination to a data source or service.

Context

Options for creating destinations in the Cloud Foundry environment.

- Create a mobile destination, configuring all aspects of its connection, including security. This gives you full control of all available configuration settings.
- Create a mobile destination using an existing Cloud Foundry service instance in the same space. This enables you to quickly configure a connection by reusing an existing instance.
- Create a mobile destination that references an existing cloud destination. The mobile destination uses the security configuration of the cloud destination. This enables you to use existing cloud destinations that are already available on the SAP BTP sub-account for a mobile application in Mobile Services.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first)
3. Select the add icon . Alternatively you can create a destination using an existing Cloud Foundry service instance in the same space, as described in [Creating a Destination with Existing Service Instances](#).



If the create icon does not appear, this means you have reached the destination limit imposed by your service plan. You may also see a message, such as "Total destinations per application can be 5 only in a free license type". To mitigate, delete one of the destinations. See [Service Plans](#) for information about service plan limits.

4. In [Create Destination](#), under [Basic Info](#), enter the following values as required, and then click [Next](#):

Field	Value
Destination Name	Provide a name for the destination.
SAP Destination service	Select to create a mobile destination that uses the settings of an existing SAP Destination service.
Cloud Destination Name (appears only if SAP Destination service is enabled)	Select the value help icon, and then select an existing cloud destination. If the icon does not appear, you must enable "Enable Access to Cloud Destination". When enabled, the icon appears. Note that the value help only shows destinations having the property <code>MobileEnabled=true</code> .

Field	Value
Standard Path to Add (appears only if SAP Destination service is enabled)	Enter the path information to be added by default to the URL configured in the cloud destination.
URL	<p>URL that the application uses to access business data on the back-end system or service. If the URL points to a service, it must include the document destination that you assign to the service.</p> <p>You can enter an "http://" URL or an "https://" URL (for the latter, you are prompted for keystore, certificate, and truststore values later in the process).</p> <div> <p>Note</p> <p>If you are implementing Custom Push, enter the URL of the push notification server that will distribute push notifications. The Mobile Services server sends a general notification message to the push destination server. The destination server handles further forwarding of the notifications. See <i>Custom Push</i> for additional information.</p> </div>
Allowed Paths	<p>Use Allowed Paths to restrict access to a few sub-paths of the Destination URL.</p> <p>For example, if the Destination URL is configured as <code>https://www.test.com/sap</code>, and you only want to allow access to <code>https://www.test.com/sap/customer.svc</code> and <code>https://www.test.com/sap/product.svc</code>, then configure Allowed Paths to contain <code>/customer.svc</code> and <code>/product.svc</code>. HTTP requests starting with these URLs will be allowed, and others will be rejected with a 403 status code.</p> <p>The entered paths are case-sensitive. Please notice that wild card characters are not supported but are implicit at the end of the string.</p>
Use Cloud Connector (does not appear if SAP Destination service is enabled)	<p>(Optional) Indicates if SAP Cloud Connector must be used.</p> <p>If you choose to use the SAP Cloud Connector, provide the location id in the <i>Cloud Connector Location ID</i> field or leave it blank.</p>

Field	Value
Maximum Connections	<p>(Optional) The maximum number of connections that this application can use for connection pooling. Valid values are 0–9999.</p> <p>Factors to consider are:</p> <ul style="list-style-type: none"> • Expected number of concurrent application users • Acceptable load for the back-end system <p>To disable connection pooling, set the value to 0. This creates a new connection for each new request, which may increase processing times. SAP recommends that you disable connection pooling only if the back-end system does not support pooled connections</p>
Maximum Request Size (bytes)	<p>(Optional) The maximum size of the HTTP request payload in bytes. Set a value from 1 – 2147483647. Set to 0 to indicate the request should not contain a request body. If the value is not set, then 10485760 (default) is used.</p>
Timeout (ms)	<p>(Optional) The number of milliseconds before the connection times out. If set to 0, a system-wide default value of 60 seconds is used.</p>
Online Request Threshold	<p>(Optional) Set a threshold value from 1 – 2147483647 to restrict the maximum number of requests per second. Leave blank (default) or set to 0 or -1 to remove a threshold.</p>

Field	Value
Rewrite Mode	<div> <div>  Note </div> <div> <p>To enable applications that use external back ends to run offline, you must select either Rewrite URL or Rewrite URL on Back End.</p> </div> </div> <p>Select one of:</p> <ul style="list-style-type: none"> Rewrite URL – in request and response messages, the mobile services replaces all back-end URLs with the mobile service URL. The Rewrite URL format for Web-type applications is <code>https://<mobileServiceHost>/<back-end_connection_ID>?X-SMP-APPID=<applicationID></code>. <div> <div>  Note </div> <div> <p>If you enable URL rewrite in the Mobile Offline service, you must also configure these settings for the application's Mobile Connectivity destination:</p> <ul style="list-style-type: none"> Set the "Rewrite Mode" attribute to "Rewrite URL". Ensure that the "Relative Rewrite Paths" attribute is empty. <p>See Editing the Application Configuration File for information about URL Rewrite in Offline Service.</p> </div> </div> <ul style="list-style-type: none"> Rewrite URL on Back End – the back end rewrites the URLs. The mobile services forwards the host name and port to the back end in an HTTP header, and the back end creates the URL to retrieve back-end resources. To expose the full URL to clients, the mobile service passes the endpoint in the X-SMP-ENDPOINTNAME header. The URL format for Web applications is <code>https://<host>/<back-end path>?X-SMP-APPID=<applicationID></code>. No Rewriting – request and response messages are not modified. The mobile services passes messages directly between clients and the back end. The URL format for Web applications is <code>https://<mobileServiceHost>/<back-end_connection_ID>?X-SMP-APPID=<applicationID></code>.

Field	Value
-------	-------

Note

- The mobile services does not provide the functionality to use No Rewriting mode to support external back ends for offline usage.
- SAP Mobile Cards: The server performs a virus check scan for the incoming data.
- *Rewrite URL*: The server performs a virus check scan for the incoming data. Rewrite URL applications should use only *No Rewriting* mode.

- *Custom Rewrite URL* – for request and response messages, you can define a search string and a replacement string, which need not be URLs.

For more details about the different rewrite mode options, see *Rewrite Modes*.

Keep X-Forwarded-* Header

This option appears when you edit a destination. Select the check box to enable or disable the `SetXForwardedHeaders` property (disabled by default). The property is used by proxy to establish endpoint connection.

Select the check box to enable or disable the option to pass along the X-Forwarded-* headers, which contain information about the sender of the HTTP request and the original URL being called (disabled by default) to the Destination.

5. (Optional) If you set the *Rewrite Mode* to *Custom Rewrite URL*, define its values on the subsequent *Inbound Rewrite Rules* and *Outbound Rewrite Rules* screens, and then click *Next*.


For more information, see *Rewrite Modes*.

6. (Optional) On *Custom Headers*, configure key/value pairs for the header destination, and then click *Next*.

Select  to configure headers for the destination.

For example, you can:

- Set up a static HTTP header for an API key when consuming SAP Business Accelerator Hub APIs.
- Create a custom header with a cookie value that includes a variable for outgoing requests. For example, if a back-end server generates a cookie, all subsequent requests for the same back end include a custom header with the value of the cookie (if the specified cookie does not exist, the custom header is not added to the outgoing request). See *Custom Headers for Cookie Variables*.

The headers must comply with IF RFC Standards, 7230, section 3.2: <https://tools.ietf.org/html/rfc7230#section-3.2> .

The key/value pairs are sent to the back end with each request.

Field	Value
Header Name	<ul style="list-style-type: none"> • Must not be empty. • Must start with an alphabetic character. • Must include only alphanumeric characters, numbers, and minus signs (no special characters).
Header Value	<ul style="list-style-type: none"> • Can be empty. • The first and last character cannot be a space, per HTTP standards. • The value format for a cookie header: "\$ {cookie::<cookie name> } ". For example, "\$ {cookie::SAP_SESSIONID_GW1_001} " which retrieves the value of cookie <"SAP_SESSIONID_GW1_001"> at runtime.
Override Client	Indicates if the header should override the header sent from client.

7. (Optional) On [Annotations](#), configure annotations for the destination, so that all apps using this destination can access the annotations and generate the UI, and then click [Next](#).

Choose [Add Annotation URL](#) if you know the URL for the annotation file. Choose [Add Annotation File](#) to browse and upload the file.

When configuring the annotation, keep in mind that the current framework is based on the Endpoint configuration. This means that the back-end URL is the base, and any path must be a relative path to the base URL, otherwise security issues may be the result.

For example, if the back-end URL is:

```
http://host:port/odata.svc/
```

and the annotation path is:

```
/a1/annotations(...)
```

the actual URL requested is:

```
http://host:port/odata.svc/a1/annotations(...)
```

Note

Relative paths are not supported when an ABAP Gateway back end, and the OData Service and annotation file are in different paths.

8. On [Destination Configuration](#), configure the following as required and then select [Next](#).

Field	Value
Relative Rewrite Paths	Enter a comma-delimited list of relative URLs, for example, /sap/bc , /sap/public/bc .

Field	Value
	<p>If an application requires data from a back end that uses relative URLs, define them here. The mobile services rewrites the relative URLs to include the connection name, enabling access to the back-end data.</p> <p>For example, a Web service application requests an HTML page named <code>abc.html</code>, which contains the relative URLs <code>/sap/bc</code> and <code>/sap/public/bc</code> in its <code>src</code> or <code>href</code> tags. When a request is made, the relative URLs contained in the response are rewritten, so that subsequent requests (to these relative URLs) can be processed correctly. For example, if "webApp" is the connection name, and the response contains the relative URLs <code>/sap/bc</code>, <code>/sap/public/bc</code>, these are changed to <code>/webApp/sap/bc</code>, <code>/webApp/sap/public/bc</code></p>
Propagate User Name	<p>(Not applicable when application Security Configuration is set to None) When enabled, the back end uses information in the <code>X-SMP-ENDUSERNAME <user name></code> header to identify the user who sent the request. See HTTP Headers Used to Propagate User IDs.</p> <p>By default, this option is disabled.</p>
Virus Scans	<ul style="list-style-type: none"> • Inbound Traffic: The server performs a virus check scan for the incoming data. Rewrite URLC2G-product-name: The server performs a virus check scan for the incoming data. Rewrite URL • Outbound Traffic: The server performs a virus check scan for the outgoing data.
SSO Mechanism (does not appear if SAP Destination service is enabled)	<p>Select a single sign-on option from the list of available options.</p>

Mobile Services supports the following SSO options:


- **None** does not add any authentication or user information to the request. It can be used when the targeted service does not require any authentication or the authentication is performed by adding an API-Key header, like `X-API-Key`.
- **Basic Authentication** is used when the target service requires technical user authentication. No caller information is forwarded.
- **OAuth2 Client Credentials** can be used for technical user authentication via OAuth2 client credentials. No caller information is forwarded.
- **Forward Authentication** can be used when the target is deployed in the same space and uses the same XSUAA instance, or the back-end grants scope access to Mobile Services. Check [Forward User Token to App Router](#) when the target is the app router.
- **OAuth2 User Token Exchange** is used when the target is in the same BTP Cloud Foundry sub account, but uses a different XSUAA instance.

- **Application to Application SSO** is used for legacy services deployed on BTP Neo. This is an interim solution and it is recommended to migrate your service to BTP Cloud Foundry.
- **OAuth2 SAML Bearer Assertion** is for legacy services deployed on BTP Neo that uses OAuth2 SAML authentication. This is an interim solution and it is recommended to migrate your service to BTP Cloud Foundry.

For detailed configuration examples, see [How to configure SSO Mechanism in Mobile Connectivity](#) 📄.

SSO Mechanism	Description
Application-to-Application SSO	<p>Enables mobile services to propagate user identities to other applications, which are consumed (deployed or subscribed) in the same SAP BTP account. A user identity is propagated to the application that is specified in the URL.</p> <ul style="list-style-type: none"> • <i>Issuer</i> – the trusted application source, such as "mobile services." • <i>Audience</i> – the recipient audience, such as "hana.ondemand.com." • <i>Signing Key</i> – the generated key used to propagate the user identity. Select Generate Key to generate the signing key. A SAML Download field appears in the destination overview page once you complete the configuration. <p>Keep in mind these requirements:</p> <ul style="list-style-type: none"> • The proxy type for the destination must be Internet. • To configure ApptoAppSSO for an application not hosted on the same SAP BTP account; see the saml2_audience section in Application-to-Application SSO Authentication.
OAuth2 SAML Bearer Assertion	<p>Enables applications to use SAML assertions to access OAuth-protected resources.</p> <p>Enter:</p> <ul style="list-style-type: none"> • Forward User Token to AppRouter – enable capability to forward user tokens to the AppRouter for SSO authentication. • Audience (required) – intended assertion audience, which is verified by the target OAuth authorization server. • Token Service URL (required) – URL of the OAuth server. • Token Service URL Type (required) – the URL type, either Dedicated or Common. • Client Key (required) – key that identifies the consumer to the authorization server. • Client Secret – password for the token service user (no longer mandatory). • SAML Assertion Issuer (required) – issuer of the SAML assertion. • Signing Key (required) – key used for signing the SAML Assertion, which is used for exchanging the token from OAuth Server. Select Generate Key to generate the signing key. Once you finish the configuration, a summary page is provided for the destination. In the Security section, the SAML Metadata field appears. You can configure the metadata expiration date for the application, and download the SAML metadata locally. • Name ID Format – value of the NameIdFormat tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication. Select a value from the drop-down list: <pre>urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified [default value] urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress</pre>

SSO Mechanism	Description
	<pre>urn:oasis:names:tc:SAML:2.0:nameid-format:persistent urn:oasis:names:tc:SAML:2.0:nameid-format:transient</pre> <ul style="list-style-type: none"> • Authentication Context – the value of the AuthnContextClassRef tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication. See the SAML 2.0 specification. • Scope – (optional) limits an application's access to a users account. You can make one or more entries; this information is presented to the user in the consent screen, and the access token issued to the application will be limited to those granted. • SAML System User – SAML user who requests an access token from the OAuth authorization server. If this property is not specified, the currently logged-in user is used. • SAML Name Qualifier – security domain of the user for which the access token is requested. • Company ID – the company identifier associated with the security domain. • User ID Source – the issuer of the user identifier, typically the currently logged-in user. • API Key – the API-key that is sent in the request header and used as the password to authenticate a request.
Basic Authentication	<p>Enables basic authentication to the back-end system.</p> <p>Enter:</p> <ul style="list-style-type: none"> • User Name and Password – the user name and password to access the back-end system. If you do not provide a user name and password, and the mobile services authenticates the end-user credentials using Basic, the user name and password credentials are propagated to the back end. • Credential Charset Name – the default is UTF-8. Use the default, or enter another value. If the destination is an SAP Net Weaver ABAP application server, you must enter ISO-8859-1. (This is because SAP Mobile Services uses UTF-8 encoding and SAP Net Weaver ABAP application server requires ISO-8859-1 encoding).
No Authentication	<p>Back ends require no credentials for authentication. Your destination is granted direct access to the relevant on-premise service.</p>
Forward Authentication	<p>Forwards the incoming JSON Web Token (JWT) in the authorization header to the backend. The token could be used to log in as a certain user type, such as an Admin. Typical uses for Forward Authentication include accessing the WeChat sample backend, and accessing the Fiori Launchpad as a user type.</p> <p>When the Forward Authentication SSO mechanism is configured for an end point, the checkbox Forward User Token To AppRouter appears. Select the checkbox to enable. When enabled, the user token is forwarded to the app-router application as an <code>x-approuter-authorization</code> header.</p>

SSO Mechanism	Description
<div>  Note The app-router version installed on the back-end server must be equal to or later than version 5.15.0. Earlier versions do not support SSO access. </div>	
Cloud Connector SSO	Enables principal propagation through SAP Cloud Connector.
OAuth2 Client Credentials	<p>The Client Credentials grant is used when applications request an access token to access their own resources, not on behalf of a user.</p> <p>Enter:</p> <ul style="list-style-type: none"> Token Service URL – URL of the OAuth server. Client ID – the client username. Client Secret – the client password. Scope – (optional) limits an application's access to a users account. You can make one or more entries; this information is presented to the user in the consent screen, and the access token issued to the application will be limited to those granted.
OAuth2 User Token Exchange	<p>Supports JSON Web Token (JWT) authentication. Token exchange enables easier integration of Cloud Foundry service instances from the same space. You can find the required information in the Service Key details of the target service. If required, you must create a Service Key beforehand.</p> <p>Enter:</p> <ul style="list-style-type: none"> Forward User Token to AppRouter – enable capability to forward user tokens to the AppRouter for SSO authentication. Token Service URL – URL of the OAuth token exchange server. Token Service URL Type – select Dedicated (default) or Common. Common is used for multi-tenant services, whereas Dedicated is used for single tenant services. Client ID – the client username. Client Secret – the client password. Scope – (optional) limits an application's access to a users account. You can make one or more entries; this information is presented to the user in the consent screen, and the access token issued to the application will be limited to those granted.

- On [Certificate Configuration](#), if you entered an "https://" URL in step 4, you are prompted to enter keystore, certificate, and truststore values. If you entered an "http://" URL, or enabled SAP Destination service, proceed to the next step.

Certificate Configuration

Field	Value
Keystore	The Keystore file in .keystore or .jks format. You can Browse to locate a keystore.
Encoded Keystore	The name for the encrypted version of your private key.
Keystore Password	The password associated with the Keystore.

Field	Value
Certificate Alias	The alias name associated with the Keystore.
Truststore	The Truststore file. You can Browse to locate a truststore.
Encoded Truststore	The name for the encrypted version of your private key.
Truststore Password	The password associated with the Truststore.

10. Click [Finish](#) to complete the configuration. A summary of configuration settings appears, with appropriate categories for the app, such as [Info](#), [Security](#), [Rewrite Method](#), [Annotations](#), and [Customer Headers](#).

You can click [Edit](#) to make corrections.

For ► [Security](#) ► [SAML Metadata](#) ►, select [Download](#) to download application-level metadata locally. In [Download Metadata](#), specify the metadata expiration date, and then select [Download](#). You can select one year (default), or use the date picker to select the expiration month and year. If the metadata value is set at a global level, this value will overwrite the global value for this application.

Once you create a new destination, action icons appear on the overview page for the selected application. Use the icons to test connectivity.

- Select [Testing OData Destination Quality in a Browser](#) to check OData quality for the selected destination. This enables administrators and developers to conduct inspections on the back-end OData service, and to identify and resolve potential issues.
- Select [OData Application Destination Test](#) to test destination links for OData applications. This enables an Admin user to verify an OData service, and provides a way to browse metadata information and preview back-end data
- Select [Ping](#) to verify the connection to the destination.
- Select [Launch in Browser](#) to test destination connectivity for all authentication types and for OData applications from a web browser. A new browser tab is launched using the mobile application URL, the same URL that the application uses to interact with the destination.

ⓘ Note

This feature may not be available for some app types. If the feature is not available for the selected app type, the action icons do not appear or are grayed out.

[Rewrite Modes \(Cloud Foundry\) \[page 103\]](#)

For destinations, the rewrite mode defines how the mobile services handles request and response messages.

[Custom Headers for Cookie Variables \[page 107\]](#)

As an administrator, you can configure a mobile destination that uses a custom header with a cookie value in outgoing request, instead of using a static HTTP header.

Related Information

[Custom Push \[page 147\]](#)

[Editing a Destination \[page 109\]](#)

[Pinging a Destination \[page 110\]](#)

1.5.5.1.3.6.1.1 Rewrite Modes (Cloud Foundry)

For destinations, the rewrite mode defines how the mobile services handles request and response messages.

Rewriting is mainly relevant for OData content, which often contains entity and other references as absolute URLs. Since the incoming URL that is used by the mobile application is different from the URL of the OData service, the content must be rewritten. The rewriting usually applies to request and response messages.

This functionality can also be useful for other content (for example HTML). If content rewriting is not desired, please choose “No Rewriting”.

The Destinations defined for an application are usually addressed by name, as you can also see in the APIs tab of the application.

Naming follows the pattern:

```
http://<server>/<destination name>
```

Note

This naming URL pattern does not apply to the “Rewrite in Backend” Rewrite mode. Please see the specific section below for details.

For Mobile Services on Cloud Foundry, the server name is unique for each mobile application and no further information is required.

In contrast, for Mobile Services on Neo, all mobile applications use the same hostname, and the configured application must be specified in the HTTP request by any of these means:

- X-SMP-APPCID cookie as retrieved during the application onboarding (this is mainly done by mobile applications based upon the SMP 3.<x> SDKs).
- X-SMP-APPID header with a value that corresponds to the application ID.
- X-SMP-APPID URL parameter with a value that corresponds to the application ID (this is mainly used by web-applications or for testing reasons).

Mobile applications that are built upon any of the supported SDKs already take care of this.

Rewrite URL

This is the default URL rewrite mode and usually the safest choice for OData content. This mode replaces the mobile services URL (including the Destination name) with the value defined as the back-end URL in all request and response messages. It also rewrites the Location header of the response accordingly.

The rewrite URL format for Web-type applications is `https://<host>/<applicationID>`.

In case there is an additional system in between mobile services and the back end, the standard URL Rewrite might not be sufficient, since all three systems are using different hostnames, and mobile services does not have knowledge about the hostname of the final destination. In this case you must use “Custom Rewrite URL”.

Rewrite URL and Performance

You can use Rewrite URL with the Mobile Connectivity and Mobile Offline components. Mobile Connectivity is recommended for OData content and Mobile Offline is recommended for offline requests, since the `no_url_rewrite` option provides better download performance.

When handled through the Mobile Connectivity component, an Offline download request can trigger several to dozens more back-end requests, each of which could involve heavy computing work because of URL rewriting. The Mobile Offline component handles URL rewriting much more efficiently, by parsing OData and only finding/replacing the OData elements that contain URLs, while the Mobile Connectivity component does not parse the payload format and ends up finding/replacing the entire payload.

Mobile Connectivity destinations have four URL Rewrite options:

1. Rewrite URL
2. Rewrite URL on Back End
3. Custom Rewrite URL
4. No Rewrite

The Offline `no_url_rewrite` option is used to perform URL rewriting if the Mobile Connectivity destination is configured with option #1 and the request comes from Offline. If a request comes directly from a device app (online request) to the Mobile Connectivity component, the Mobile Connectivity component performs the URL rewrite as configured. Offline cannot perform a proper URL rewrite if Mobile Connectivity is configured with option #3, and does not derive any performance benefit if Mobile Connectivity is configured with options #2 or #4.

❗ Note

If you enable URL Rewrite in the Mobile Offline component, you must also configure these settings for the application's Mobile Connectivity destination:

- Set the "Rewrite Mode" attribute to "Rewrite URL".
- Ensure that the "Relative Rewrite Paths" attribute is empty.

See *Editing the Application Configuration File* for information about [URL Rewrite in Offline Service](#).

Rewrite URL on Back End

The back end rewrites the URLs. The mobile services forwards the host name and port to the back end as an HTTP header, and the back end creates the URL to retrieve back-end resources. Mobile Services does not alter the content, except for the Location header in the response.

This requires Mobile Services to not alter the path of the URL in any way and for this reason a Destination that is configured for Rewrite URL in Back End can be called by the Mobile Application not by the Destination name but by the path of the back end. For example:

- Back-end URL – `https://ldcigm3.wdf.sap.corp:50057/sap/opu/odata/sap/FINCUSTFACTSHEET/`
- URL exposed to clients – `https://<host>:<port>/sap/opu/odata/sap/FINCUSTFACTSHEET/`

To expose the full URL to clients, the mobile services passes the request path completely:

- Back-end URL – `http://ldcigm3.wdf.sap.corp:50057/sap/opu/odata/sap/FINCUSTFACTSHEET/`
- URL exposed to clients – `http://<host>:<port>/sap/opu/odata/sap/FINCUSTFACTSHEET/`
- URL format for Web applications – `https://<host>/<back-end path>`, for example:

```
https://mobiletest-xxxx.new.ondemand.com/sap/bc/ui5_ui5/ui2/ushell/shells/abap/FioriLaunchpad.html
```

Considerations

- The base path of the URL must correspond to the path of the back-end URL. For other rewrite modes, the base path must contain the Destination name (as in the example above).
- If you change the value to or from [Rewrite URL on Back End](#), inform the application developer, who must update the application base URL accordingly, for both online and offline mobile application scenarios.
- If you change the rewrite value, you must also reconfigure the mobile application.

Rewrite via Cloud Platform App

To enable requests to fetch data from HTML5 applications that are hosted on SAP BTP, select [Rewrite via Cloud Platform App](#). This sends the host information in the `X-FORWARDED-FOR` header, and HTML5 applications send it to back-end systems in the `Host` header.

- If selected, the host name is sent to the back end in the `X-FORWARDED-FOR` HTTP header.
- If not selected, the host name is sent to the back end in the `Host` HTTP header.

No Rewriting


Request and response messages are not modified; they are sent directly between clients and the back end.

Note

The mobile services does not provide the functionality to use No Rewriting mode to support external back ends for offline usage.


Custom Rewrite URL

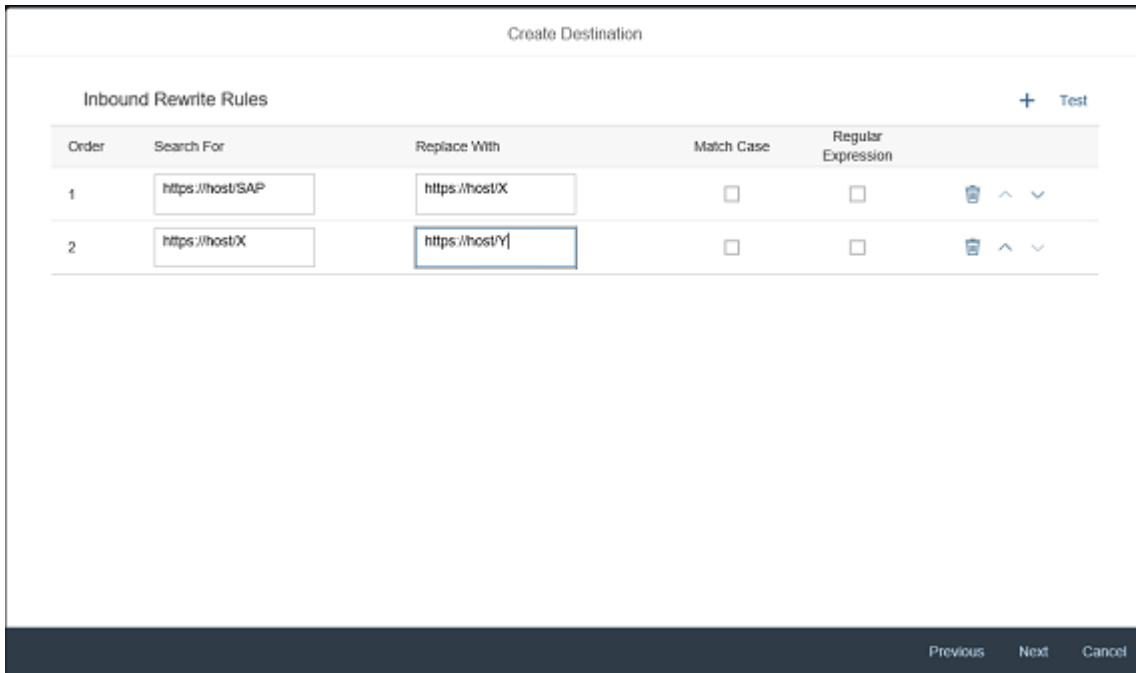
For request and response messages, you can define a search string and a replacement string, which need not be URLs. Clients initiate incoming messages, which pass through the mobile services and terminate in the back-end system. Outgoing messages travel in the opposite direction.

If you select [Custom Rewrite URL](#), click [Next](#). In the Create Destination dialog, click the Add icon . On two separate screens, define the Inbound Rewrite Rules, and then the Outbound Rewrite Rules:

- [Search For](#) – string to find. To facilitate searching, you can use placeholder variables.
- [Replace With](#) – replacement string. For example, you can convert an absolute URL to a relative URL by replacing "http://host" with an empty string.

- [Match Case](#) – whether the case of the search string must match exactly.
- [Regular Expression](#) – allow regular expressions in the strings.

To define another rewrite pair, click the Add icon , and define the properties listed above. If you define more than one, you can sort them to change the order. The system works through the definitions, from top to bottom, searching for a Search For string that matches the input string from the client. For example, assume you define two rewrite pairs, in this order:



The screenshot shows a 'Create Destination' window with a section titled 'Inbound Rewrite Rules'. There are two rules listed in a table. Rule 1 has 'Search For' as 'https://host/SAP' and 'Replace With' as 'https://host/X'. Rule 2 has 'Search For' as 'https://host/X' and 'Replace With' as 'https://host/Y'. Both rules have 'Match Case' and 'Regular Expression' checkboxes unchecked. To the right of the table are icons for adding, deleting, and sorting rules, and a 'Test' button. At the bottom of the window are 'Previous', 'Next', and 'Cancel' buttons.

Order	Search For	Replace With	Match Case	Regular Expression	
1	https://host/SAP	https://host/X	<input type="checkbox"/>	<input type="checkbox"/>	
2	https://host/X	https://host/Y	<input type="checkbox"/>	<input type="checkbox"/>	

For this example, the system receives the input string "https://host/SAP" from the client, which matches the Search For string in the first definition, so it replaces the input string with "https://host/X." The system then looks at the next rewrite definition, compares "https://host/X" with the Search For string, finds a match, and replaces this string with "https://host/Y"; this is the output string.

If the definitions are in the reverse order, and the system receives the input string "https://host/SAP," the output string would be "https://host/X"; moving top to bottom, the matching definition is the last one.

To test a Custom Rewrite URL configuration for either inbound or outbound rules:

1. Select the configuration, and click [Test](#).
2. In the Test Rewrite Rules dialog, enter the [Input](#) string.
3. Click [Rewrite](#). The replacement string appears under [Output](#).

To edit or delete a configuration, select it, and click the appropriate icon.

1.5.5.1.3.6.1.2 Custom Headers for Cookie Variables

As an administrator, you can configure a mobile destination that uses a custom header with a cookie value in outgoing request, instead of using a static HTTP header.

Prerequisites

Identify the cookie value that the back-end server generates, such as "SAP_SESSIONID_GW1_001".

Context

With this feature you can specify a cookie name as the custom header value; retrieve the cookie value from the server; and use the custom header value at runtime. For example, a back-end server generates a cookie named, "SAP_SESSIONID_GW1_001". Once this cookie is generated, all requests that are sent to this back-end server include the custom header "SAP_SESSIONID" and the cookie value, "SAP_SESSIONID_GW1_001".

The value format is "\${cookie::<<cookie name>>}", for example, "\${cookie::SAP_SESSIONID_GW1_001}". This retrieves the cookie value "SAP_SESSIONID_GW1_001" at runtime. If the specified cookie does not exist, the custom header is not added to the outgoing request.

Procedure

1. Create a mobile destination using the [Create Destination](#) dialog, as described in *Creating a Destination*.
2. On the [Custom Headers](#) page, for [Header Name](#) and [Header Value](#), create a key/value pair using the custom header and cookie values.
3. Complete additional configuration, and then save the destination.

Next Steps

Once configured, all subsequent outgoing requests that are sent to the back-end server include the custom header and cookie value.

1.5.5.1.3.6.2 Creating a Destination with Existing Service Instances

Define a new destination to a back-end system using existing Cloud Foundry service instances.

Prerequisites

For Document service:

- In SAP BTP, *Entitlements*, add an entitlement for the *Document Management, Repository option* to the subscriber subaccount.
The *Document Management, Repository option* entitlement must include a quota. The free plan includes a quota of two units. If that is not enough, you can update it by removing the current entitlement and creating a new entitlement with a larger quota.
- In SAP BTP, *Services* > *Service Instances*, create a *Document Management, integration option* instance for the service instance.
- Only Document service instances that have been allow-listed are available.

Context

You can create destinations from existing service instances that are available in the same Cloud Foundry space. All aspects of the destination are configured, including URL and SSP (usually OAuth Token Exchange). You can select only one service instance at a time, so if you want to create multiple service instances you must create separate destinations.

Note

Currently these service instances can be integrated:

- Workflow service instances
- Document service instances

Procedure

1. In Mobile Services cockpit, select *Mobile Applications* > *Native/MDK*.
2. Select an application, then select *Mobile Connectivity* under *Assigned Features* (or add it first).
3. Select *Use a Cloud Foundry Service*.
4. On *Select Cloud Foundry Service*, select a service from the list of available service instances, and select *OK*.
When the document service destination is created successfully, you can *Ping* it.

You can only add one service at a time. Depending on the service, one or several destinations are created..

5. You can take action, such as edit or delete; or you can add another destination using another existing service. For some SSO methods, you can test the destination.

1.5.5.1.3.6.3 Editing a Destination

Modify settings for an existing destination.


Context

📘 Note

To prevent momentary inconsistencies, SAP recommends that you modify destination configurations when few users are active. Users should be able to use destinations without inconsistencies after you save the changes.

In Mobile Services cockpit, you can view the properties of Fiori applications and connections that were developed using other tools and imported into SAP Mobile Services, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK** or **SAP Mobile Cards**.
2. Select an application, then select **Mobile Connectivity** under **Assigned Features** (or add it first)
3. Select a destination and click .
4. In the **Edit Destination** window, edit the details as required.

📘 Note

If the application is configured with an origin policy, some fields may not be available.

5. Click **Finish**.

1.5.5.1.3.6.4 Pinging a Destination

Test whether a destination is accessible.

Context

Keep in mind that the ping option is not available for SAP Destination service, or for some applications. One exception is that if Forward Authentication is used in combination with endpoint address names that end with `/SampleServices/ESPM.svc/v2` or `/SampleServices/ESPM.svc/v4`, the ping test is available.

If you do not see the ping icon, or it is grayed out, ping is not supported for the destination or application. In this case you can test the destination in a web browser. See *Testing Destinations in a Browser* for information.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ► or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first)
3. Select a destination and click [Ping](#).

You see either:

- [Ping Successful](#) – connection is accessible.
- [Ping Failed](#) – connection is not accessible. Click [Show Details](#) to see more information.

Related Information


[Testing Destinations in a Browser \[page 114\]](#)

1.5.5.1.3.6.5 Deleting a Destination

You can delete a destination only if it is not mapped to an application.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ► or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first)

3. Select a destination and click .
4. Click [OK](#) to confirm. You are prompted if the destination is in use and cannot be deleted.

1.5.5.1.3.6.6 Configuring Application-to-Application SSO Authentication

The application-to-application single sign-on (SSO) authentication type allows SAP Mobile Services to propagate the identity of the logged-in user to another application, which is consumed (deployed or subscribed) in the same SAP BTP account.

Prerequisites

- The application that is receiving the SSO operation (the receiving application) must be consumable from the same SAP BTP account – either by being deployed or through a subscription.
- The receiving application can be either a Java or an HTML5 application.
- When developing your own Java application, see [Enabling Authentication](#) for information about enabling the application to accept application-to-application SSO.
Be sure to configure your account to allow principal propagation. For more information, see [ID Federation with the Corporate Identity Provider](#), the "Specifying Custom Local Provider Settings" section.

Note

This setting is account specific, which means that if set to Enabled, all applications within the account accept principal propagation.

Context

The user identity is propagated to the application specified in the URL, which you can configure in Mobile Services cockpit.

Procedure

1. Define a new application.
2. Select the application, and add Connectivity under Assigned Features as described in *Managing Application Features*.
3. Create or assign a destination that includes the Application-to-Application SSO mechanism to the connectivity.

1.5.5.1.3.6.7 Testing OData Destination Quality

OData destination quality testing ensures data accuracy and reliability. It is essential for ensuring that OData destinations are functioning properly and delivering high-quality data for use in various applications and systems.

Prerequisites


The Mobile Offline feature must be enabled for the application.

Context





OData quality refers to how well an OData service is supported by the Mobile Offline feature. A new browser tab is launched to perform the OData quality check and display the results.

Note

You may find it useful to enable the `Allow Omitting Max Length` property in the application configuration file. When enabled, no maximum length is imposed. See *Editing the Application Configuration File*, the "endpoint properties" step.

It is not possible for OData quality checks to identify issues that could only be detected through Create, Update, Delete (CUD) operations. This limitation may be revisited once the new feature, [Dry-run of data modification requests](#) , is implemented in OData V4.02.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first)
3. Under [Actions](#), click the Checking OData Quality of Destinations in a Browser icon . This launches a new browser tab, using a modified destination URL.
4. In the [Welcome](#) screen, log in using your SAP ID user name and password.

Upon successful authentication, the destination's [General Info](#) data appears in the browser window.

Note

If you check the browser URL, you will see that the URL has been changed and `?auth=uaa` has been appended. This indicates the destination was tested through the App-router.

5. Under [OData Quality Check Results](#), you can review the quality check results and recommendations for the selected application.

Quality Check Properties

Property	Description
Code	An identifier for reference.
Severity	The impact of the condition (High, Medium, Low). Note that issues with High severity prevent Offline synchronization from succeeding, if the reported entity set is used.
Description	A detailed explanation of the issue.
Solution	If the description is not informative enough, this column provides additional information on what needs to be done by the administrator or developer to resolve the issue. This could involve making adjustments to the back end, the Offline configuration, and so forth.
References	Provides links to all the relevant documentation, help, notes, and so forth.

6. Close the web browser window when ready. Use the quality check to make changes to ensure the destination performance.

Related Information

[Editing the Application Configuration File \[page 122\]](#)

1.5.5.1.3.6.8 Testing an OData Destination

You can test destination links for OData applications from the Connections window.

Prerequisites

The destination's SSO Basic Auth must be configured with a technical user. If the icon does not appear, the OData application destination may not be configured correctly, or may not be supported for the SSO method used.


Context

This test tool enables an Admin User to verify an OData service, and provides a way to browse metadata information and preview back-end data.

If you do not see the OData application test destination icon, or it is grayed out, it may indicate that testing is not supported for the destination.

If Allowed Paths is enabled for the destination, you can select one of the paths.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ► or *SAP Mobile Cards*.
2. Select an application, then select *Mobile Connectivity* under *Assigned Features* (or add it first)
3. Under *Actions*, click the OData application test destination icon .

If Allowed Paths is enabled when the test tool opens, the first Allowed Path is set by default and shown after loading. You can select one of the options from the *Relative OData Service Path* list, and select *Get Data*. If connected, the data appears in the *Entity Data* section. If not connected, "No data is being received from the server" appears instead.

4. View the basic information about the application and destination. You can enter the relative OData service path of the destination URL, and check the OData metadata definition.
5. View the OData service entities, the properties and data of the entities; and the JSON raw data and XML raw data of each entity.

1.5.5.1.3.6.9 Testing Destinations in a Browser

You can test destination connectivity for all authentication types and for OData applications, from a web browser.


Context

In this scenario, a new browser tab is launched using the mobile application URL, the same URL that the application uses to interact with the destination. The URL is modified (the parameter, `?auth=uaa`, is appended), and the request is routed through App-router. This is the same way that a mobile application that is used by an end user interacts with the destination.

If Allowed Paths is enabled for the destination, you are prompted to select one of the paths.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ► or *SAP Mobile Cards*.
2. Select an application, then select *Mobile Connectivity* under *Assigned Features* (or add it first).

3. Select a destination and under **Actions**, click . If Allowed Paths is enabled for the destination, the tool tip indicates "Launch Allowed Path in Browser"; otherwise the tool tip indicates "Launch in Browser". This launches a new browser tab, using a modified destination URL.

If Allowed Paths is enabled, select one of the options from [Select an Allowed Path](#).

4. In the **Welcome** screen, log in using your SAP ID user name and password.

Upon successful authentication, the destination's data appears in the browser window. You can retrieve entity data.

Note

If you check the browser URL, you will see that the URL has been changed and `?auth=uaa` has been appended. This indicates the destination was tested through the App-router.

5. Close the web browser window when ready.

1.5.5.1.3.6.10 Accessing Cloud Destinations

You can configure Cloud Foundry applications to use cloud destinations that are defined in SAP BTP for the customer subaccount.

Prerequisites

- The application must have been developed with, or migrated to, the single Mobile Services model, as described in *Upgrading Apps to Mobile Services Cloud Foundry Service*.
- Cloud destinations are defined for the customer subaccount, typically by someone who is a global account member or a subaccount security administrator.
- You must set the additional property `MobileEnabled` to "true", to use the cloud destination for Mobile Connectivity.





To learn more, see the *SAP Cloud Platform Connectivity* guide, specifically:

- [Access the Destinations Editor](#)
- [Create HTTP Destinations](#)

Context

This feature provides a way to enable cloud destinations for mobile applications on SAP Mobile Services Cloud Foundry. When the proxy server receives an incoming request in the format of `/destination_{<destination name>}/{<path>}`, and access to cloud destination is enabled, the cloud destination is located and the configured data is used to send the request securely to the back end.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#).
View a list of mobile applications, with columns for Application ID, Name, Vendor, State, Outdated, and Creation Date.
2. Select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first).
3. Under [Cloud Destinations](#), select  .

Next Steps

Once enabled, you can use the cloud destinations that are defined in `/destination_{<destination name>}/{<path>}` when configuring connectivity for the application.

1.5.5.1.3.7 Tracing Network Activity

Trace network activity between mobile services, apps and other services, and download tracing information in a `.zip` file.




Prerequisites

Configure the trace settings before you start recording.

Context



Record HTTP traffic for incoming and outgoing requests between mobile clients, apps, and other services. Developers, testers and administrators can use these recordings to inspect the network traffic and isolate issues.

Procedure


1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#).
2. Select an application, then select [Mobile Network Trace](#) under [Assigned Features](#) (or add it first).
3. Select the [Configuration](#) tab.
4. Under [New Recording](#), select recording options.

By default, SAP Mobile Services traces all users and content types for the selected application, but you can filter on specific activity instead.

- [User Name](#) – perform network tracing for a specific user.
 - [Content Types](#) – perform network tracing for a specific content type.
 - [Duration](#) – set a time duration during which to run the tracing session. A tracing session automatically stops once it has run for the duration you specify.
 - [Log Message](#) – choose the content to record:
 - [Log Message Header](#): Record only entry headers (default).
 - [Log Message Header and Message Body](#): Record both the header and body for each message.
5. Once you've specified the options, click [Start Recording](#). Network tracing starts, using these options.
6. Under [Recordings](#), you see a list of all the network traces that have been recorded during the last seven days.

Columns	Description
Start Recording Date	The date and time (UTC+ 0000) from which the recording begins.
End Recording Date	The date and time (UTC+ 0000) at which the recording ends
User Name	The user name for the account that has recorded the network trace.
Content Types	The media type selected for the network trace recording.
Log Message	The log message type, whether Log Message Header or Log Message Header and Body.
Action	There are three essential actions you can perform on the recorded network trace information: <ul style="list-style-type: none">• Stop: Stop a network trace that is in progress.• Delete: Click . Delete appears instead of Stop when a recording is completed.• Download: To download a specific network trace, click .

You can also:

- Click  to sort network trace recordings using column names.
- Search for a specific network trace recording.
- Select [Delete All](#) to delete all trace logs.

Note

Network trace data that is older than seven days is deleted automatically.

1.5.5.1.3.8 Editing JSON Storage

Enable and manage persistent JSON storage for the selected application.

Context

The application configuration that comes with an app downloaded from a global app store, is written to an application-level storage service during onboarding. In a similar fashion, user- and device-level configuration information is written to separate storage services during onboarding. The administrator can manage the storage services from the cockpit, which includes assigning read and write roles for the application storage, and customizing JSON storage code.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#) or [SAP Mobile Cards](#)
2. Select an application, and then [Mobile Settings Exchange](#) under [Assigned Features](#). Navigate to the [JSON Storage](#) tab.

Under [Storage Level](#) you can view three levels of storage in a tree format – application, user, and device.


```
Application Level
  User Level
    Device Level
    Device Level
  User Level
    Device Level
  User Level
    Device Level
    Device Level
    Device Level
```

When you select a tree node, the storage JSON appears in the [JSON Editor](#) to the right. Keep in mind:

- The tree only lists users that use JSON storage, or that have one or more registered devices that use JSON storage.
 - The tree lists all user devices, whether the device has storage JSON.
3. (Optional) Under [Storage Level](#), use the filters to adjust the storage entries you see for the application. You can limit or expand the number of entries to retrieve, search for a user by name, or search the tree for a specific value. You can expand or collapse the tree of storage entries.
 4. Select a level in the tree to manage storage for the application.

Storage Service	Description
Application Level	View and modify application-level storage.
User Level	View and modify user-level storage.
Device Level	View and modify device-level storage.

5. (Optional) Select an Application Level object to manage the storage service at the application-level. The application ID appears, as well as a [Role Definition](#) section.

- a. For each JSON Object level, assign the read and write roles required. Enter one role name at a time followed by [Enter](#) to create a list of roles.
 - b. In [JSON Editor](#) you can preview code in JSON format, or modify the JSON format for customized properties. Use [Beautify](#) to format code, and [Full Preview](#) to view the full application level JSON storage for customized properties.
6. (Optional) Select a User Level object to manage storage service at the user-level. The user ID appears.
In [JSON Editor](#) you can preview code in JSON format, or modify the JSON format. Use [Beautify](#) to format code, and [Full Preview](#) to view the full user level JSON. The full JSON is a combination of user level and application level JSON.
7. (Optional) Select a Device Level object to manage the storage service at the device-level. The device ID appears.
In [JSON Editor](#) you can preview code in JSON format, or modify the JSON format. Use [Beautify](#) to format code, and [Full Preview](#) to view the full device level JSON. The JSON is a combination of device level, user level, and application level JSON.
8. (Optional) Select an object, and click  to remove it.
9. Click [Save](#) at any time to save your changes, or [Reset](#) to discard all unsaved changes.

1.5.5.1.3.9 Defining Offline Settings for Applications

Define offline settings for the selected application. Offline support enables client applications to access back-end data without a connection. When offline, applications access data from an offline store on the client. SAP Mobile Services moves data between the back end and the client offline store.

Context

The destination settings determine how SAP Mobile Services creates the initial offline store on the client, and how it processes requests for updates from the back end. Define offline back-end connection settings for an application by importing a configuration (`.ini`) file that has been prepared by a developer. To update the offline configuration, you can either:

- From Mobile Services cockpit, select the edit button to edit the offline configuration and save it.
- Remove the current configuration, update the configuration file, and then reimport the file. Before updating this file, confer with a developer.


See these sections in the *Native OData App Development*, SAP SDK for iOS, and SAP SDK for Android documentation:

- [iOS Applications](#)
- [Android Applications](#)
- [Windows Applications](#)
- [Developing Offline Applications](#)

Mobile Services offline supports key properties that were skipped over or missed when default values were generated by the back-end OData service (V4 only), including object entries, and referential constraint

definitions that contain partial primary keys. For primary keys that are not referenced by foreign keys, a server-side default value is used. This enables Mobile Services to process Cloud Application Programming (CAP) draft entities for offline. See [Understanding Keys](#) for more information.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#).
2. Select an application, then select [Mobile Offline Access](#) under [Assigned Features](#).
3. Select the [Configuration](#) tab to see a list of configured destinations.
4. To import an offline configuration, select the  icon, browse and select a configuration file.

Note

Only `.ini` files can be imported. When you import settings, the state of the offline configuration changes to Configured.


For information about Application Configuration files, see:

- SAP Mobile Platform SDK apps – [Defining an Application Configuration File with Defining Requests](#)
 - SAP SDK for iOS apps – [Defining an Application Configuration File with Defining Queries](#)
 - SAP SDK for Android apps – [Defining an Application Configuration File with Defining Queries](#)
5. Select the [Offline Policies](#) tab to set incoming request throttling threshold and offline store upload policies for the application.
 - a. Under [Throttling Policy](#), select [Enable Throttling](#) to set throttling policy. Once the policy is enabled, you can control the activity level for the application. Set thresholds for throttling activity once thresholds are reached.

For [Incoming Request Throttling Threshold](#), configure the request throttling threshold, from 1 to 200 requests. The default value is 200 requests per second. When the offline service reaches the threshold value, no additional requests are handled.
 - b. Under [Offline Store Upload Policy](#), select [Enable Offline Store Upload](#) to set offline store upload policies. Once the policy is enabled, the device user can upload the local offline store files to the server for the developer to analyze or troubleshoot, subject to the upload policy settings.

Specify policy values:

Offline Store Upload Policy Properties



Property	Default	Description
Delete Offline Store After	7 days	The time to elapse before the offline store files are deleted automatically. This security measure protects the device user, but be sure to allow enough time for the developer to perform the troubleshooting or analysis.
<div>  Note The maximum value is 30 days. For legacy configuration values that are more than 30, the value is changed to 30 automatically. </div>		
Maximum Offline Store Size	32 MB	The maximum size allowed for the offline store files.

6. Select the [Offline Stores](#) tab to view the uploaded offline stores.

You see the list of existing offline stores that are available in offline settings. You can also view the Unique ID, Device ID, Store Name, Notes, Created By, File Size, Creation Data, and the Actions you can perform for each offline store. You cannot modify these values, but they are useful for finding and sorting offline stores.

Offline Stores

Properties	Descriptions
Unique ID	The unique identifier for the offline store.
Device ID	The device identifier associated with the offline store.
Name	The offline store name.
Note	Text notes about the offline store, up to 120 characters.
By	Creator of the offline store.
File Size (KB)	The current offline file size in KB.
Creation Date (UTC+0800)	The offline store creation date in UTC format.
Actions	Actions that can be taken, such as deleting the offline store.

7. To download an existing offline store, click the  icon.
8. To delete an existing offline store, select the corresponding check box, and click the  icon.
9. Select [Save](#).
10. Select [Info](#) to see feature details, such as useful URLs.

[Editing the Application Configuration File \[page 122\]](#)

Use the Offline editor to make updates to the application configuration file.

1.5.5.1.3.9.1 Editing the Application Configuration File

Use the Offline editor to make updates to the application configuration file.

Context

You can view base endpoint properties for the selected offline application, and set some custom parameters in the application configuration file using Mobile Services cockpit.

An example of a custom parameter is to disable Cross-Site Request Forgery (CSRF) for the development and test environment, using the `disable_fetch_csrf_token_request` property described in step 5. Create a key-value pair using `Y` to disable CSRF Token protection. This is not supported in a production system because of security reasons, but is useful in a QA, development, or test system.

Another example of a custom parameter is to enter before and after defining requests at the end-point level, using the `before_function` and `after_function` properties described in step 5. The offline service sends separate requests to the back-end server based on this configuration. You could further customize by implementing the ' `{syncType}` ' parameter in the destination URL to specify different actions for an initial sync and a delta sync.

To learn more, see [Application Configuration File \(iOS\)](#) and [Application Configuration File \(Android\)](#).


Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, then select [Mobile Offline Access](#) under [Assigned Features](#) (or add it first).

On the [Configuration](#) tab, any offline data store destinations are listed.


Destination Properties

Property	Description
Destination	The offline client endpoint destination, such as <code>com.<domain>edm.sampleservice.v2</code> .
State	The current state of the destination, such as No Custom Settings or Configured.
Action	The action to take such as Create, Edit or Delete.

3. Select the  icon.
4. Specify endpoint properties, and then click [Next](#).

Endpoint Properties

Property	Description
Destination Name	(Display only) The destination name cannot be modified.

Property	Description
Prepopulates Offline Data	Whether to prepopulate offline data on the client. Select Yes, No or Shared-only. If you select No, then in the analytics chart, the average initial download time will be less than the average refresh download time.
Data Refresh Interval	How often to refresh the data on the client when online, in minutes.
Service Document Format	The acceptable service document format to use for the OData back-end server. You can enter a customized value as a string, such as <code>application/name</code> , or you can select <code>application/atomsvc+xml</code> , <code>application/xml</code> or <code>application/json</code> .
OData Communication Format	The format used, such as <code>application/json;q=1</code> , <code>application/atom+xml;q=0.5</code> .
Delta Communication Format	A subset of the OData communication format, such as <code>application/atom+xml</code> .
Database Collation	Character set for database collation, such as <code>UTF8BIN</code> .
Database Case Sensitivity	Whether database case matters; select to enable case sensitivity.
JSON Date Time Offset	The date-time offset format, such as UTC or Defined by Offset Portion.
Local Data Expiration	How long to keep local data before it expires, in hours.
Allow Omitting Max Length	Whether to limit the maximum length of an entry; select to impose no maximum length.
<div>  Note Consider enabling this property if you are using the Checking OData Destination Quality feature. See <i>Testing OData Destination Quality</i> for information. </div>	
Content ID Header Location	Content identifier header, such as Mime or Operation.
Number of Max Delta Resends	The maximum number of resends allowed for changes, such as 5.
Batch All Defining Requests	Whether to batch all requests when online.

Property	Description
Delta Tracking	<p>Whether to use Mobile Services to send delta responses to the client even if the back end can't support delta query. Delta tracking can help to decrease the data transfer size during the sync; in subsequent syncs, Mobile Services only sends added/updated/deleted data to the client instead of sending all data tracking changes.</p> <ul style="list-style-type: none"> • Auto (default) - indicates that Mobile Services should determine automatically whether to use the middleware delta tracking that is provided by Mobile Services, or to use back-end delta tracking: <ul style="list-style-type: none"> • If the back-end server supports delta query, Mobile Services uses the back-end delta query and does not enable middleware delta tracking. • If the back-end server does not support delta query, Mobile Services enables middleware delta tracking to support delta tracking using Mobile Services. • Always - indicates that Mobile Services should enable middleware delta tracking even if the back-end server supports delta query. This option is not recommended for most scenarios. It should be used only when the back-end server's delta query is not very reliable. For example, the back-end server returns the whole data set even when it receives a delta query to track changes only. • Never - indicates that Mobile Services should not enable middleware delta tracking. When the server supports delta query, disabling middleware delta tracking can help decrease memory use and database operations, and help improve sync performance. This option is not recommended when the back-end server does not support delta query.
Download Threads	<p>Enables you to download offline data using multiple threads. This can speed up download for large datasets. Set between 1-3. Multi-thread upload is enabled if the value is bigger than 1. Leave blank or set to 0 if multiple threads are not used.</p> <p>See <i>Configuring Offline Multi-thread Sync</i> for information.</p>

Property	Description
Full Download if Metadata Changes	Specify whether to download the full OData offline schema to the device on sync if the metadata changes. The parameter is enabled by default, meaning if you sync from the device, the full database is downloaded. Remove the checkmark to disable the feature, meaning if you sync from the device, only changes are downloaded to the device.
URL Rewrite in Offline Service	Specify whether to enable URL rewrite for the selected offline application. Your selection sets the <code>no_url_rewrite</code> property. If you select the checkbox (sets the property to <code>true</code>) offline URL rewriting is enabled (default). If you leave the checkbox unselected (sets the property to <code>false</code>) offline URL rewriting is disabled, and the Mobile Connectivity proxy service handles URL rewriting, if it is configured.

Note

If you enable URL rewrite in the Mobile Offline service, you must also configure these settings for the Mobile Connectivity destination:

- Set the "Rewrite Mode" attribute to "Rewrite URL".
- Ensure that the "Relative Rewrite Paths" attribute is empty.

See *Creating a Destination* for information about these attributes.

5. In *Endpoint Customized Properties*, click  to add private parameters, and then click *Next*.

View the list of key-value pairs, and create or edit key-value pairs for the parameters.

Endpoint Customized Properties

Property	Description
Key	The key, such as <code>disable_fetch_csrf_token_request</code> . Select a key from the list or input a valid key directly. Keys are categorized by type, such as String, Integer, Integer Array, and Boolean. Once you select a key, select the help button for more information, such as <code>Enter Y or N to disable or enable fetching CSRF token for requests; if not set the default is N.</code>
Value	The key value, such as <code>Y</code> . The field provides a hint as to the value that is expected.


Property	Description
Action	The action to take, such as Create or Delete.

Endpoint Customized Property Descriptions

Key	Description	Type
before_function	<p>Enter before_function to defining requests at the end-point level; the offline service sends separate requests to the back-end server based on this configuration.</p> <p>Example format: mat: GetProductsByRating? rating=3</p> <p>Optionally you can add ' {syncType} ' as a placeholder to the destination URL:</p> <p>Example format: before_function = OpenSyncSession? SAPProductTechName= ' SAP_SERVICE_ASSET_MANAGER ' & SyncType= ' {syncType} '</p>	String
after_function	<p>Enter after_function to defining requests at the end-point level; the offline service sends separate requests to the back-end server based on this configuration.</p> <p>Example format: mat: GetProductsByRating? rating=4</p> <p>Optionally you can add ' {syncType} ' as a placeholder to the destination URL:</p> <p>Example format: after_function = CloseSyncSession? SAPProductTechName= ' SAP_SERVICE_ASSET_MANAGER ' & SyncType= ' {syncType} '</p>	String

Key	Description	Type
offline_db_creation_parameters	Specifies a list of client database creation options, semicolon separated. Example format: precision=100;scale=10	String
media_link_prefix	Enter a prefix to use for media links. Example format: test2	String
repeatable_requests_lifetime	Controls how long in minutes that repeatable request responses are kept. The default value is 72000 minutes (50 days). Example format: 36000	Integer
max_length_facet_default	Specifies the default MaxLength values in bytes that should be assigned by the system when they are not provided in the OData metadata. If not specified, the default value is 1536 bytes. Example format: 2048	Integer
max_retry_times	Enter an integer value between 0 to 10 to establish the maximum number of times Mobile Services should try to connect to the back-end server, when the back-end server cannot be accessed. For example, enter 5 for five retries. The range is 0 - 10, but a mid-range value is recommended. Example format: 5	Integer
repeatable_exclude_status	Specifies a list of status codes which should not be checked for repeatable requests. Example format: 401 , 403	Integer Array

Key	Description	Type
check_repeatable_requests	<p>Enter Y or N to enable or disable checking repeatable requests; if not set the default is N. When set to Y, repeatable request checking is enabled and Mobile Services does not send repeated request headers to the back-end server.</p> <div> <p>Note</p> <p>If the <code>send_standard_repeatability</code> property is also set to Y, it will be overridden, so should be set to N.</p> </div>	Boolean
download_in_order	Indicates whether defining queries should be downloaded in the order in which they are defined in the configuration file. If not set the default is N.	Boolean
disable_fetch_csrf_token_request	Enter Y or N to disable or enable fetching CSRF token for requests; if not set the default is N.	Boolean
force_medialink_absolute_url	Enter Y or N to enable or disable forcing medialink absolute URL; if not set the default is Y.	Boolean
skip_nullablerelationship_check	Enter Y or N to enable or disable skipping nullable relationship check; if not set the default is Y.	Boolean
allow_defining_query_remove	Enter Y or N to enable or disable defining query removal; if not set the default is N.	Boolean
generate_implicit_entity_id_in_response	Enter Y or N to enable or disable generating implicit entity id in response; if not set the default is N.	Boolean

Key	Description	Type
v4_use_post_http_method_or_patch (default is false) v2_use_post_http_method_or_patch (default is true)	<p>These options control the HTTP method used by the server to interact with the OData back end to replay a "PATCH" operation. The mechanism is similar to what is described in OData Operations (OData Version 2.0) , see <i>Section 3.2, Method Tunneling through POST</i>.</p> <p>When the option is set to true, the server sends a POST request with the header "X-HTTP-Method: MERGE". Otherwise, the server sends the "PATCH" directly. How you set these options depends on whether or not the OData back end supports the PATCH method. If not set, the default is used.</p> <p>For example, enter Y or N to enable or disable the POST tunneling method for OData.</p>	Boolean
disable_normalize_time	<p>Enables Mobile Services to keep the original values of EDM .Time type attributes that are sent from the back-end server instead of normalizing them. Enter Y or N (the default). When set to N, Mobile Services normalizes the values.</p> <p>For example, if the value of an EDM .Time type attribute is PT01D08M20S, Mobile Services normalizes it to PT1D8M20S and saves it to a varchar type, so it affects the column sort. When set to Y, Mobile Services retains the original values from the back-end server, without normalization.</p>	Boolean


Key	Description	Type
<code>add_csrf_token_header_for_get_request</code>	Enables you to add X-CSRF-Token headers to GET requests. Its value can be Y or N (the default is N). When set to Y, Mobile Services adds a X-CSRF-Token header to the GET requests sent to the back-end server; if set to N, no X-CSRF-Token header is added to GET requests.	Boolean
<code>early_populate_backend_generated_values</code>	Indicates whether missing values can be populated from the back-end server during upload, if available. Currently, this feature only supports entity creation and does not support entity patching or updating. The default is N.	Boolean
<code>log_request_headers_when_bad_request</code>	Indicates whether to log request headers. The default is N. When set to Y, Mobile Services logs the request headers when meeting bad requests. If not set the default is N.	Boolean
<code>wait_shared_cache_updated</code>	Enter Y to indicate that when a shared defining query is synchronized, and Mobile Services finds that the defining query cache is being updated, the current sync waits until the cache is updated, in order to download the latest data. If not set the default is N.	Boolean
<code>not_populate_refconstraint</code>	<p>Disable sending referential constraint data to the client. When set to "Y", Mobile Services does not populate the client with referential constraint data during the initial download. The benefits are that the database size is decreased and the initial download performance is improved, but query processing may take a little longer on the client.</p> <p>If not set, the default is N. You can also change the setting from "Y" to "N" to pause the behavior temporarily.</p>	Boolean

Key	Description	Type
<code>ignore_stream_request_4xx_error</code>	<p>Temporarily ignore some 4xx errors. By design, when the client sets a <code>definingQuery</code> with <code>automaticallyRetrievesStreams=true</code> and the back-end server returns a 4xx code for some streams, the sync between client and back-end server terminates.</p> <p>When set to "Y", Mobile Services ignores these 4xx errors and lets the sync continue, loading the other streams instead of breaking the sync. The option is useful when back-end media entities are not stable, and the media content is not immediately required on the device after the initial download. One limitation is that the ignored media content will not be downloaded again until its entity is updated (some properties are changed or its <code>etag</code> is changed).</p> <p>If not set, the default is "N". You can also change the setting from "Y" to "N" to pause the behavior temporarily.</p>	Boolean
<code>send_original_repeatability</code>	<p>When set, the client sends repeatable requests in accordance with the original (2013) specification. Set "Y" (the default) to indicate repeatable requests should be sent using the 2013 specification. Set to "N" to indicate that the 2013 standard is not used.</p> <div> <p>Note</p> <p>When the <code>check_repeatable_requests</code> and <code>send_original_repeatability</code> properties are both set to Y, the <code>send_original_repeatability</code> property is ineffective and should be disabled.</p> </div>	Boolean

Key	Description	Type
send_standard_repeatability	When set, the client sends repeatable requests in accordance with the standard (2020) specification. Set to "Y" to indicate repeatable requests should be sent using the 2020 specification. Set to "N" (the default) to indicate that the 2020 standard is not used.	Boolean

Note

When the `check_repeatable_requests` and `send_standard_repeatability` properties are both set to Y, the `send_standard_repeatability` property is ineffective and should be disabled.


6. In [Client Indexes](#), click  to add client database index parameters, and then click [Next](#).

The index parameters are for the client database that is used for offline storage. This enables you to configure specific actions for a client database entity, such as sort data in ascending or descending order.

Client Index Properties

Property	Description
Entity Type Name	Enter the client database <code>EntityType</code> name, in the format <code><namespace>.<EntityType></code> . For example, <code>Sample.Customer</code> .
Properties	Enter one or more property names, which will be used to create the <code>EntityType</code> index, in the format <code>Property1A[ASC DESC][, Property1B[ASC DESC][, ...]</code> . The default sort order is ascending (ASC), if no entry is made. For example, <code>FirstName , Nick</code> . The client index properties.
Action	The action to take, such as delete.


See [Application Configuration File](#) for more information about the `indexed_type` property.

7. In [Defining Requests](#), click  to add the defining requests parameters, and then click [Next](#).

Request Properties

Property	Description
Name	The defining request name.
Refresh Interval (min)	The interval time, in minutes, between downloads of the shared data.
Delta Tracking	<p>Whether to use Mobile Services to send delta response to the client even if the back end can't support delta query. You can specify <i>Auto</i>, <i>Always</i>, or <i>Never</i> as described for <i>Delta Tracking</i> in step 4.</p> <div> <p>Note</p> <p>If you specified a <i>Delta Tracking</i> value for <i>Endpoint Properties</i> (step 4), the same value is automatically used for <i>Defining Request</i>. You can leave blank to use the same value, or you can specify a different value for <i>Defining Requests</i>.</p> </div>
Token Lifetime (min)	<p>The time, in minutes, until the OData delta token expires. This value controls how long the oldest change log data are retained.</p> <p>While valid, only changed data is downloaded to the offline store. When the delta token expires, the entire data set is replaced. The default is 21600 minutes (about 15 days). This value is recommended for optimal database usage and performance of the mobile service.</p>
OData Communication Format	<p>Specify the format to use for OData payload non-delta requests at the defining query level. For example, some back ends such as Gateway require that you use XML and not JSON. With this property you can configure an efficient defining query with a JSON content type.</p> <p>You can select one of the pre-defined query options:</p> <ul style="list-style-type: none"> <code>application/atom+xml</code> <code>application/json</code> <p>Or you can write your own. Be sure to set a value if you write your own, or it will be left empty. If no configuration is provided, the destination level configuration is used.</p>

Property	Description
Delta Communication Format	<p>Specify the format to use for OData payload delta requests at the defining query level. For example, some back ends such as Gateway require that you use XML and not JSON. With this property you can configure an efficient defining query with a JSON content type, even if you're using a Gateway back end without delta support. You can also configure a Gateway back end with delta support using XML.</p> <p>You can select one of the pre-defined query options:</p> <ul style="list-style-type: none"> <code>application/atom+xml</code> <code>application/json</code> <p>Or you can write your own. Be sure to set a value if you write your own, or it will be left empty. If no configuration is provided, the destination level configuration is used.</p>
Share Data	Whether data is shared among different clients or users.
Download Order	<p>If you enabled multiple thread download in step 4, you can assign a number to each defining request to indicate the order each request should be processed. Lower numbers are processed before higher numbers. For example, if you assign 66 to one request and 777 to another, 66 will be processed before 777.</p> <p>See <i>Configuring Offline Multi-thread Sync</i> for information.</p>
Action	The action to take, such as Delete.

8. In *Defining Request Groups*, click  to add any defining request groups, and then click *Finish*.

Request Groups Properties

Property	Description
Request Names	Select one or more defining requests.
Action	The action to take, such as Delete.

[Configuring Offline Multi-thread Sync \[page 135\]](#)

You can now configure multiple back-end threads in Mobile Services cockpit for offline synchronization. This gives you more control over synchronization, leading to better sync performance, especially for large data sets.

1.5.5.1.3.9.1.1 Configuring Offline Multi-thread Sync

You can now configure multiple back-end threads in Mobile Services cockpit for offline synchronization. This gives you more control over synchronization, leading to better sync performance, especially for large data sets.

Prerequisites

The application must support multiple-thread sync in the imported offline configuration file (.ini), and the feature must be enabled for the tenant and its customers.

Context

Once you upload the offline configuration file (.ini), configure the application to download offline data using multiple threads. First, enable Download Threads for the application, and then assign a Download Order value to each defining request. The download requests are processed in phases, from the lowest value first, to the highest value. For example, a value of 66 is processed before 777. Each phase must be completed before the next phases starts.

Note

Requests in the same download phase are downloaded in parallel, with no guaranty for the download order of the requests within the phase. For example, requests in download phase 66 are all handled in parallel. If download order matters, specify a different download phase for them.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, then select **Mobile Offline Access** under **Assigned Features** (or add it first).
3. On the **Configuration** tab, create or edit an offline configuration, using information in *Editing the Application Configuration File* for entries.

In the **Offline Configuration** wizard, the **Endpoint Info** page, make an entry in **Download Threads** to enable multi-thread synchronization. You can enter 1, 2, or 3 for download threads. Multi-thread download is enabled if the value is larger than 1. If you leave the field blank or enter 1, multi-thread sync is not enabled. Select **Next** to proceed.

4. In **Define Request**, make an entry in **Download Order** for each defining request to indicate the order the request should be processed. Lower numbers are processed before higher numbers. For example, if you assign 66 to one request and 777 to another, 66 will be processed before 777. Select **Next** to proceed.
5. When you are finished with these entries, select **Finish** to save. The offline configuration is updated.

The next offline synchronization will be processed in the order you specified.

Related Information

[Editing the Application Configuration File \[page 122\]](#)

1.5.5.1.3.10 Using the Mobile Sample OData ESPM

A sample OData service is available for developers to use during development and testing. The sample OData service also lets you evaluate how delta tokens are handled in your test application. The sample back end should not be used for production.

Prerequisites

- The Developer role.
- For the Developer role, the runtime root URL is the application service URL to which the sample OData service is binding, for example, `<application_service_url>/SampleServices/ESPM.svc/OData_version`.
- Click the metadata URL to view the sample OData entities and their metadata.

Context

Administrators can view the sample back-end service via the cockpit. View the root service and metadata URLs, and generate sample sales orders and purchase orders for multiple entity sets. View the data for each entity in a separate text file, and reset the sample data.

When developers add the *Mobile Sample OData ESPM* feature to an application, the *Mobile Connectivity* feature is also added, if it has not already been added, and a destination named "com.sap.edm.sampleservice.v4" is automatically created in *Mobile Connectivity*. Sample OData can be consumed via the *Mobile Connectivity* service, as well as be consumed directly via the Sample OData runtime URL (for example, `<application_service_url>/SampleServices/ESPM.svc/OData_version`).

Delta tokens let you retrieve the changes that have been applied to a service. You can send a request to the back end and the delta token is appended to the response.

By default, the application uses the following settings to provide a security check, and enable the developer to access the service using the following URLs:

- V4: `https://<application_service_url>/com.sap.edm.sampleservice.v4`
- V2: `https://<application_service_url>/com.sap.edm.sampleservice.v2`

For the SSO mechanism, Forward Authentication SSO is used, and cannot be modified.

Once you add the service to an application, you can use the service via the cockpit. You can also view the root service and metadata URLs, view the data for each entity in a separate text file, and reset the sample data. You can generate a demo HTTP error if a quantity value of `SalesOrderItem` is below zero. Use the OData error handling API to show the error for your application.








The demo error handling of OData sample service is to add a data validation functionality for the *quantity* field of *SalesOrderItem*. When you try to create or update the quantity value using the POST operation, which is less than or equal to 0, the response is:

```
<?xml version="1.0" encoding="utf-8"?>
<error xmlns="http://docs.oasis-open.org/odata/ns/metadata">
<code>400</code>
<message>DataServiceException: The value of quantity should be greater than
zero.</message><innererror>
<details>com.sap.xscript.data.DataServiceException: The value of quantity should
be greater than zero.</details>
</innererror>
</error>
```

The Products entity includes sample images identified, using the property "PictureUrl". You can view the images in the cockpit, and provide image examples for tutorials. You can change the image examples.

```
EntityType Name="Product">
<Key>
<PropertyRef Name="ProductID"/>
</Key>
<Property Name="Category" Type="Edm.String" Nullable="true" MaxLength="40"/>
<Property Name="CategoryName" Type="Edm.String" Nullable="true" MaxLength="40"/>
<Property Name="CurrencyCode" Type="Edm.String" Nullable="true" MaxLength="5"/>
<Property Name="DimensionDepth" Type="Edm.Decimal" Nullable="true"
Precision="13" Scale="4"/>
<Property Name="DimensionHeight" Type="Edm.Decimal" Nullable="true"
Precision="13" Scale="4"/>
<Property Name="DimensionUnit" Type="Edm.String" Nullable="true" MaxLength="3"/>
<Property Name="DimensionWidth" Type="Edm.Decimal" Nullable="true"
Precision="13" Scale="4"/>
<Property Name="LongDescription" Type="Edm.String" Nullable="true"
MaxLength="255"/>
<Property Name="Name" Type="Edm.String" Nullable="false" MaxLength="80"/>
<Property Name="PictureUrl" Type="Edm.String" Nullable="true" MaxLength="255"/>
<Property Name="Price" Type="Edm.Decimal" Nullable="true" Precision="23"
Scale="3"/>
<Property Name="ProductID" Type="Edm.Int64" Nullable="false" MaxLength="10"/>
<Property Name="QuantityUnit" Type="Edm.String" Nullable="true" MaxLength="3"/>
<Property Name="ShortDescription" Type="Edm.String" Nullable="true"
MaxLength="255"/>
<Property Name="SupplierID" Type="Edm.Int64" Nullable="false" MaxLength="10"/>
<Property Name="Weight" Type="Edm.Decimal" Nullable="true" Precision="13"
Scale="3"/>
<Property Name="WeightUnit" Type="Edm.String" Nullable="true" MaxLength="3"/>
<Property Name="Picture" Type="Edm.Stream" Nullable="true"/>
<NavigationProperty Name="Supplier" Type="ESPM.Supplier" Nullable="false"
Partner="Products">
<ReferentialConstraint Property="SupplierID" ReferencedProperty="SupplierID"/>
</NavigationProperty>
<NavigationProperty Name="Stock" Type="ESPM.Stock" Nullable="true"
Partner="Product">
<OnDelete Action="Cascade"/>
</NavigationProperty>
<NavigationProperty Name="PurchaseOrderItems"
Type="Collection(ESPM.PurchaseOrderItem)" Partner="Product">
<OnDelete Action="None"/>
</NavigationProperty>
<NavigationProperty Name="SalesOrderItems"
Type="Collection(ESPM.SalesOrderItem)" Partner="Product">
<OnDelete Action="None"/>
</NavigationProperty>
</EntityType>
```

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#) > [Native/MDK](#) .
2. Select an application.
3. To use the sample OData service, in the [Info](#) tab, click the  icon in the [Assigned Features](#) section.
4. In the [Add feature](#) window, select [Mobile Sample OData ESPM](#) and click [OK](#).
5. Select the [OData Version, V4](#) (the default) or [V2](#).
6. Select the [Entity Sets](#).
7. Select an operation for the selected entity set.
 - To open or save a text file of the JSON file contents for the selected entity, such as `Customers.json`, click the  icon.
 - To generate ten sample sales orders (each click generates ten more), click the  icon.
 - To generate ten sample purchase orders (each click generates ten more), click the  icon.
 - To reset the data to an initial state of ten records, click the  icon.
8. Select a different entity set from the list, such as Customers, Product Categories, Product Texts, Products, and so on.

You see a table of properties for the selected entity.

Entity Properties

Property	Description
Property	Using the new OData framework with its new functionality. The new serviceList of entity properties, such as City, Country, and CustomerId.
Data Type	The property's data type, such as Edm.String, or Edm.DateTime.
Nullable	Whether the property is nullable, typically true or false.
Property Type	The property type, such as Simple.

Next Steps

Once the service is configured, the developer can access it using the following URLs after onboarding:

- V4: `com.sap.edm.sampleservice.v4`
- V2: `com.sap.edm.sampleservice.v2`

Note

The sample OData service automatically assigns the sample OData destination named `com.sap.edm.sampleservice.sample_service_version` when you add a sample back-end feature

to a mobile application. The sample back-end feature allows you to use the configuration user interface to maintain or configure back-end data. The `com.sap.edm.sampleservice.sample_service_version` takes care of the runtime connectivity channel from device to OData back end.

[Viewing Sample Pictures \[page 139\]](#)

You can preview embedded sample back-end images in Mobile Services cockpit.

[Launching the Demo App \[page 140\]](#)

Launch a simple Web user interface for the sample service from Mobile Services cockpit. This demo tool provides developers with a way to see data changes for a mobile client on the desktop, and to generate mobile apps from the annotations to understand how the mobile client works.

1.5.5.1.3.10.1 Viewing Sample Pictures

You can preview embedded sample back-end images in Mobile Services cockpit.

Prerequisites

- Log into Mobile Services cockpit with a Developer role.
- Include the [Mobile Sample OData ESPM](#) in the application definition.
- You must provide the right security information to preview images.

Context

You can preview images in several ways:

- Direct way: `https://<mobile_service_host>/<PictureURL_value>`, where the Picture URL is the complete address to access the image.
- If the [Mobile Sample OData ESPM](#) feature has been assigned to an application, you can modify the `PictureUrl` value, and preview the image.
- If the [Mobile Sample OData ESPM](#) feature has been assigned to an application and a registration has been generated, you can view the `PictureUrl` with the `X-SMP-APPCID` in the request header.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Sample OData ESPM](#) under [Assigned Features](#) (or add it first)
3. Select [Products](#) from [Entity Sets](#), and click view data .

Open the download to see the Products data, including the Picture Url values for all products in the catalog.

```
{
  "d": {
    "results": [
      {
        "Category": "Graphic Cards",
        "CategoryName": "Graphic Cards",
        "CurrencyCode": null,
        "DimensionDepth": "35.0000",
        "DimensionHeight": "17.0000",
        "DimensionUnit": "cm",
        "DimensionWidth": "22.0000",
        "LongDescription": "Proctra X: PCI-E GDDR5 3072MB",
        "Name": "Proctra X",
        "PictureUrl": "/imgs/HT-1070.jpg",
        "Price": "70.900",
        "ProductId": "6041F167-63B2-45EC-BB35-471DDEDCB3A7",
        "QuantityUnit": "EA",
        "ShortDescription": "Proctra X: PCI-E GDDR5 3072MB",
        ...
      }
    ]
  }
}
```

4. Use the PictureUrl value to preview the picture, using an available option.
5. (Optional) If the [Mobile Sample OData ESPM](#) feature has been assigned to an application, you can modify the PictureUrl value, and preview the image. Use the format: `<cockpit_admin_url>/app/<app_name>/service/sample-odata-v2-espm/v4/imgs/<picture_name>`.

1.5.5.1.3.10.2 Launching the Demo App

Launch a simple Web user interface for the sample service from Mobile Services cockpit. This demo tool provides developers with a way to see data changes for a mobile client on the desktop, and to generate mobile apps from the annotations to understand how the mobile client works.

Context

Keep in mind:

- The list page only shows the simple type entity attribute. Navigation type entities (which link to another entity) and structure type entities are not supported.
- Binary type is not supported.
- Product creation is not supported because of an XScript OData v2 limitation.
- When creating the SalesOrderItem, the SalesOrderHeader related information is also needed.
- For single object creating and editing, only necessary attributes are allowed.
- The user interface follows standard Fiori Elements behavior. You can upload its customization annotation as well, and the demo application will be rendered using the customized annotation.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#).
2. Select an application, and then under [Assigned Features](#) select [Mobile Sample OData ESPM](#).
3. In the [Configuration](#) tab, make sure [Version 4](#) is selected from the [OData Version](#) drop-down list.
4. In [Entity Sets](#), select one of the support sets, which includes Products, Customers, SalesOrderHeaders, and SalesOrderItems.
5. Select the Launch Demo app icon.
6. Log into the Sample Service Demo, based on your application security configuration. The standard screen appears for the entity you selected, such as Customers.

You can use the demo app as described in *Using the Mobile Sample OData ESPM*, except for exceptions and limitations described in *Context*.

1.5.5.1.3.11 Defining Push Notifications

Configure push-related settings for the selected application.

Use native push to send push notifications. The push listener service provided with SAP Mobile Services allows back-end systems to send native notifications to devices. Application developers must enable push notification code in applications. You can also use the native push mechanism to push notifications to a subset of users.

Note

Text message (SMS) support is not available.

For the selected application, select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first).

- Select [Configuration](#) to configure push notifications.
- Select [Push Registrations](#) to view current registrations and send push notifications. See *Managing Push Notifications* to send a message.
- Select [Service Keys](#) to view and manage API keys. See *Service Keys* to manage service keys.
- Select [Info](#) to view feature details and to download mobile service data in JSON format.

Firestore Cloud Messaging Canonical IDs

For Firestore Cloud Messaging (FCM) clients, canonical IDs prevent problems that could occur if a client application inadvertently triggers multiple registrations for the same device; for example, the device could receive duplicate messages.

If a FCM client application sends a message that contains an old registration ID, FCM processes the request and inserts the canonical ID into the `registration_id` field of the response.

SAP Mobile Services:

- Replace the old registration ID that is stored for the device with the canonical ID.

- Use the canonical ID for sending messages to the device.
- Log a customer event to inform the client of updated or deleted device registrations that result from managing canonical registration IDs.

[Predefined Push Configuration \[page 143\]](#)

(Fiori Client) Enable or disable preconfigured push settings.

[Android Push Notifications \[page 143\]](#)

To enable client applications to receive Firebase Cloud Messaging (FCM) notifications, configure Android push notifications.

[Apple Push Notifications \[page 144\]](#)

Configure APNs push notifications for the selected iOS client application or SAP Mobile Cards.

[Windows Push Notifications \[page 145\]](#)

To enable the back-end servers connected with SAP Mobile Services to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

[Baidu Push Notification \[page 146\]](#)

Baidu provides a push notification routing service for users in some countries/regions. To enable this feature, configure Mobile Services to connect.

[Custom Push \[page 147\]](#)

Custom push provides a push notification routing service to a custom push server for users in some COUNTRY/REGIONS. To enable this feature, configure a destination URL to the custom push server.

[Browser Notification \[page 149\]](#)

To enable client applications to receive browser notifications enable W3C notifications.

[Enabling Push for Subscribers \(SaaS\) \[page 152\]](#)

You can enable push notifications to be sent to app subscribers.

[Using SAML Attribute as Alternative User Name \[page 152\]](#)

You can enable the selected application to use the SAML attribute as the User Name for push notifications. This could be useful if the back end uses a different SAML Attribute to identify users and the user name is not available in the back end.

Related Information

[Managing Push Notifications \[page 233\]](#)

[Managing Push Notifications \[page 233\]](#)

[Service Keys \[page 303\]](#)

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)

[Defining Connectivity \[page 89\]](#)

[Enabling Applications to Discover Configurations \[page 179\]](#)

[Managing Applications \[page 210\]](#)

1.5.5.1.3.11.1 Predefined Push Configuration

(Fiori Client) Enable or disable preconfigured push settings.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select ► [Features](#) ► [Push Notification](#) ►
3. Under ► [Predefined Global Push Configuration](#) ► [Predefined for](#) ►, select a pre-defined push configuration when using one of the official SAP apps. Leave it blank when using your own APNs or Android push configuration.

ⓘ Note

Predefined push configuration is only supported for the app store versions for iOS and Android.

- [SAP Fiori Client for B2B](#) – push configuration for SAP Fiori Client for B2B.
 - [SAP Mobile Services Client](#) – push configuration for the client of the SAP Mobile Development Kit.
 - [SAP Asset Manager](#) – push configuration for SAP Asset Manager.
4. Click [Save](#).

1.5.5.1.3.11.2 Android Push Notifications

To enable client applications to receive Firebase Cloud Messaging (FCM) notifications, configure Android push notifications.

Prerequisites

You can use the HTTP v1 API, which uses a Firebase Service Account Private Key as the credential; or the legacy deprecated HTTP API with Sender ID and Server Key.

ⓘ Note

Mobile Services cockpit currently supports both methods.

Google has deprecated the legacy HTTP API that uses Server Key authentication and will discontinue the authentication in June 2024. You must update your configuration to Firebase Service Account Private Key beforehand.

Procedure

1. In Mobile Services cockpit, select *Mobile Applications*.
2. Select an application, then select *Mobile Push Notification* under *Assigned Features* (or add it first); for SAP Mobile Cards select ► *Features* ► *Push Notification* ►
3. Under Android select whether to use *Firebase Service Account Private Key* or *Server Key* for push credentials.

For *Firebase Service Account Private Key*:

1. In *Service Account Private Key File*, browse to select the private key file that you downloaded from the Firebase platform.
2. If the file is valid, the remaining fields are populated automatically using values in the private key file.

Firebase Private Key Properties

Properties	Description
Project ID	The project ID named in the private key file.
Private Key ID	The private key ID named in the private key file.
Client Email	The client email named in the private key file.

For *Server Key (Deprecated)*:

Server Key Credentials

Properties	Description
Access Key	The access key you obtained for your Google API project (https://firebase.google.com/docs/cloud-messaging 🔗).
Sender ID	The project identifier.

4. Click *Save*.

1.5.5.1.3.11.3 Apple Push Notifications

Configure APNs push notifications for the selected iOS client application or SAP Mobile Cards.

Context

You can use either certificate or token-based authentication for APNs. The advantage of using token-based authentication is that the credentials do not expire automatically, whereas APNs certificates usually expire after one year. You can manually control the lifecycle of the key used for token-based authentication in developer.apple.com 🔗 . Obtain the credentials used for token-based authentication from your developer account on developer.apple.com 🔗 .

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select [Features](#) [Push Notification](#).
3. To configure APNs for a development and testing environment, select [Sandbox](#) or [Production](#). See [Apple Developer - Certificates](#) for certificate types and how to retrieve.
 - Select [Sandbox](#) if your mobile app is signed with an iOS Development Certificate.
 - Select [Production](#) if your mobile app is signed with an iOS Distribution Certificate.
 - Select [None](#) if you don't want to receive APNs push notifications.
4. Choose either [Certificate](#) or [Token-Based](#) as the authentication type.

For [Certificate](#):

1. Select [Browse](#) to navigate to the APNs certificate file, select the file, and click [Open](#).
2. Enter a valid password.
3. (Optional) Identify the topic bundle ID that is specific to the iOS application you're setting up to receive notifications. If not configured, the bundle ID is extracted from the certificate.

For [Token-Based](#):

Token Properties

Property	Description
Topic (Bundle ID)	The bundle ID that is specific to the iOS application you're setting up to receive notifications.
Team ID	The 10-character team ID you use for developing your company's apps. Obtain this value from your developer account.
Key ID	The 10-character string that is provided by Apple when you create a key.
Key (P8)	The authentication token signing key (.p8) file that you've downloaded from Apple. Select Browse to navigate to the APNs key file, select the file, and click Open .

5. Save your changes.

1.5.5.1.3.11.4 Windows Push Notifications

To enable the back-end servers connected with SAP Mobile Services to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select [Features](#) [Push Notification](#)
3. Under [WNS](#), enter the application credentials, which are provided by the application developer.

Property	Description
Package SID	Package security identifier
Client Secret	Client secret information

4. Click [Save](#).

1.5.5.1.3.11.5 Baidu Push Notification

Baidu provides a push notification routing service for users in some countries/regions. To enable this feature, configure Mobile Services to connect.

Prerequisites

Subscribe to the Baidu service and obtain Baidu account credentials that include an API key and a secret key.

Context

Because Google services are not available in some countries/regions, SAP uses Baidu to send push notifications to users. Mobile Services supports two Baidu push modes:

- Notification Mode – the message is delivered to the system tray (default).
- Transparent Mode – the application determines what to do with the message. The application must include a parameter setting in the push back-end API that indicates transparent mode should be used.

To trigger a transparent message for Baidu push, add the property `"msgType": 0` to the message body. The Mobile Services server reads the properties and encodes them, and then calls the Baidu push server and uses the properties to push messages to clients.

```
{
  "users": ["user id"],
  "notification": {
    "alert": "*** 消息内容 baidu message from MS ***",
    "badge": 1,
    "sound": "sound",
    "priority": "priority",
    "data": "
    {\"key\": \"value\"}
  "
}
```

```

",
"sendAsSms": false,
"apns": null,
"gcm": null,
"wns": null,
"baidu": {
  "android": {
    { "title": "**** 消息标题 Title 1 ****", "description": " ",
      "notification_builder_id": 1, "notification_basic_style": 6, "open_type": 1,
      "url": "http://developer.baidu.com", "pkg_content": "pkgContent" }
    ,
    "ios": {
      { "alert": "ios_alert", "sound": "ios_sound", "badge": 5 }
    ,
    "msgType": 0
  }
}
}
}

```

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select [Features](#) [Push Notification](#)
3. Under Baidu, select [Enable push to Android devices](#) or [Enable push to iOS devices](#). You can push to both Android and iOS devices simultaneously by selecting both the options.
4. Specify the [API Key](#) and [Secret Key](#).
5. Save your changes.

Push messages are accepted, but always queued with a `Message was queued` message, and not received on the device. The server and client use the same environment ([Sandbox](#) or [Production](#)), but the configured topic is for another environment. The APNs service rejects messages when the server certificate has been used to send multiple invalid messages. Please generate a new certificate to use.

1.5.5.1.3.11.6 Custom Push

Custom push provides a push notification routing service to a custom push server for users in some COUNTRY/REGIONS. To enable this feature, configure a destination URL to the custom push server.

Prerequisites

- The custom push server must implement a Custom Push API that is able to receive push notification messages from Mobile Services, and then coordinate distribution of push notifications to mobile clients.
- In Mobile Services cockpit, you must set up a destination URL to point to an API endpoint address, such as `https://custom.push.<server>.<host>/push`, as described in [Creating a Destination](#).

See the [GitHub repository](#)  for Custom Push API details.

Context

The Mobile Services server sends a general notification message to the push destination server. The destination server handles further forwarding of the notifications.

See the [GitHub repository](#) for example custom push micro apps and test code. The custom push examples have been tested for Huawei and Xiaomi devices.

Procedure

1. In Mobile Services cockpit, select *Mobile Applications*.
2. Select an application, then select *Mobile Push Notification* under *Assigned Features* (or add it first); for SAP Mobile Cards select **Features** **Push Notification**.
3. Under *Custom Push*, select *Enable Custom Push*.
4. In *Custom Push Destination*, select the destination that you set up for the custom push server.
5. Save your changes.

Results

Once custom push is configured with the destination URL, the device application can register its push ID, specific to local push vendors with mobile services, and mobile service stores the ID transparently.

Later, the back end calls mobile services to send push notifications to those devices. Mobile services finds the registered push ID based on the username, application ID, and/or other information like APNs and GCM, and then sends push IDs and push messages to the custom push provider.

In this process, mobile service tries not to modify the push messages, and passes the messages to the custom push provider for parsing and processing.

Related Information

[REST API Application Development Overview \[page 323\]](#)

[Push Overview \[page 36\]](#)

[Creating a Destination \[page 92\]](#)

[Accessing Services Through Proxy URLs \[page 434\]](#)

1.5.5.1.3.11.7 Browser Notification

To enable client applications to receive browser notifications enable W3C notifications.

Context

W3C enables push notification service for Google Chrome, Mozilla Firefox and Microsoft Edge browsers. To know

how these push notifications can be integrated in a web page, see [W3C Push API and Mobile Services](#) ➡

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select ► [Features](#) ► [Push Notification](#) ►
3. Under W3C, select [Enable W3C Push API for Google Chrome, Mozilla Firefox, and Microsoft Edge browsers](#)

ⓘ Note

When you disable W3C notification it removes all the previous browser subscriptions and this cannot be reverted.

4. Save your changes.

1.5.5.1.3.11.7.1 W3C Push API and Mobile Services

Mobile Services supports sending push notifications to Google Chrome, Mozilla Firefox, and Microsoft Edge (only for versions based on Chrome, available since January 2020) browsers using the Push API.

Context

First create an application or use an existing application and make sure the Mobile Push Notification feature is enabled. To enable Mobile Push Notifications see [Browser Notification \[page 149\]](#). You should then create a service key and assign the role **push_single** to it. This key is later used to send a push notification to the user.

Client code

We will now see the concept of Service Workers and the actual subscription call.

To know more about Push API, see https://developer.mozilla.org/en-US/docs/Web/API/Push_API ➡

Retrieve applicationServerKey

The applicationServerKey is required for the Java Script code to successfully subscribe for push messages. Mobile Services provides a general purpose API to retrieve the **pushId**, which in the case of W3C Push API corresponds to the base64 encoded public key of the server.

Sample Code

```
fetch('/mobileservices/push/v1/runtime/applications/dummy/
pushconfigurations/os/w3cpushapi/pushid').then((response) => {
  if (response.status >= 400 && response.status < 500) {
    return response.text()
      .then((responseText) => {
        console.log('Failed web push response: ', response,
response.status);
      });
  }
  else {
    response.json().then((pushid) => {
      this._applicationServerKey = pushid.pushId;
    });
  }
});
```

Send Subscription to Mobile Services

Once the subscription is done and the PushSubscription is available, the PushSubscription details need to be sent to Mobile Services as the pushToken. It is recommended to register the token with a Device ID to allow the same user to register several browsers for push and update and to delete the registration individually. For this reason a random value is generated and persisted into local storage.

In case there is already a registration (from a previous load of the web page), the server will return a 409 (conflict) to the POST request, which causes the request to be repeated as a PUT, to update the existing user information:

Sample Code

```
var deviceId = localStorage.getItem("mobile-device-id");
if (!deviceId) {
  deviceId = Math.floor(Math.random() * 1000000000).toString();
  localStorage.setItem("mobile-device-id", deviceId)
}
// try to register the browser - if ther is a conflict we will update the
existing record
fetch('/mobileservices/push/v1/runtime/applications/any/os/w3cpushapi/
devices/' + deviceId, {
  method: 'POST',
  cache: 'no-cache',
  headers: {
    'Content-Type': 'application/json'
  },
  body: JSON.stringify(body) })
}).then((response) => {
  if (response.status == 409) {
```

```

        fetch('/mobileservices/push/v1/runtime/applications/any/os/w3cpushapi/
devices/' + deviceId, {
            method: 'PUT',
            cache: 'no-cache',
            headers: {
                'Content-Type': 'application/json'
            },
            body: JSON.stringify(body) })
        }).then((response) => {
            console.log('Push registration update response: ', response,
response.status);
        });
    } else {
        console.log('Push registration response: ', response,
response.status);
    }
});

```

Delete Subscription from Mobile Services

In case the user revokes the privilege for push messages, the registration should be deleted:

Sample Code

```

var deviceId = localStorage.getItem("mobile-device-id");
// Remove the subscription from Mobile Services
fetch('/mobileservices/push/v1/runtime/applications/any/os/w3cpushapi/
devices/' + deviceId, {
    method: 'DELETE',
    cache: 'no-cache'
}).then((response) => {
    console.log('Push registration delete response: ', response,
response.status);
});

```

Sending the push message

You will need the API Key and URL from the Service Key that you created earlier. After replacing these two parameters and filling in the username of the registered user you can run this simple curl command to send out a push message:

Sample Code

```

curl -H 'x-api-key: API_KEY_FROM_PUSH_SERVICE_KEY' -H 'content-
type: application/json' --data '{"users":["YOUR_USERNAME"],"notification":
{"alert":"Hello"}}' URL_FROM_PUSH_SERVICE_KEY/mobileservices/push/v1/backend/
applications/any/notifications/users

```

(Depending on the OS, the notification message might not show up if the browser and web page is in the foreground)

The same command will also send push message to Android and iOS applications registered for the same user (assuming that push is correctly configured in Mobile Services).

1.5.5.1.3.11.8 Enabling Push for Subscribers (SaaS)

You can enable push notifications to be sent to app subscribers.

Context

For Native/MDK apps managed in a one-app cockpit (SaaS), the subscriber administrator can enable or disable push notifications to be sent to app subscribers. When turned off, other push configurations, such as global push or push provider, are not available.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select ► [Features](#) ► [Push Notification](#) ►.
3. Make sure the [Configuration](#) tab is selected.
4. Select [Enable Mobile Push](#) to enable the subscribers to receive push notifications created in [Push Registrations](#) (see [Sending Push Notifications](#)). Mobile push notifications are sent to subscribers
5. Select [Disable Push Notifications](#) to turn off sending push notifications. Mobile push notifications created in [Push Registrations](#) are not sent to subscribers.

Related Information

[Managing Push Notifications \[page 233\]](#)

1.5.5.1.3.11.9 Using SAML Attribute as Alternative User Name

You can enable the selected application to use the SAML attribute as the User Name for push notifications. This could be useful if the back end uses a different SAML Attribute to identify users and the user name is not available in the back end.

Context

Mobile Push Notification retrieves notification targets using a user name as the default identifier, as described in [Sending Notifications via the REST API](#). As an alternative, you can choose to use a SAML Attribute instead,

using the [Push User Name](#) option. You can specify any user attribute with a single list item, for example: email, family_name, given_name OR SAPUID.

This value is stored during push registration or update in the push registration for this device.

ⓘ Note

All existing device push registrations must be modified or deleted after modifying the [SAML Attribute Name](#) configuration. Registration or updating a registration on behalf is not supported, because SAP Mobile Services calls the XSUAA service in the users context to retrieve the SAML attribute value.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first).
3. Under [Push User Name](#), select [Use SAML Attribute as User Name](#).
4. In [SAML Attribute Name](#) enter the attribute to use, such as family_name.
5. Save your changes.

1.5.5.1.3.12 Configuring Mobile Augmented Reality

(Native/MDK only) Use the Mobile Augmented Reality feature to manage client augmented reality resources that can be accessed from mobile applications. For example, a training or museum app may be annotated to provide access to detailed information.

Prerequisites

This feature requires:

- The Version 2 Mobile Services cockpit (Version 1 is not supported).
- The native/MDK application must be configured for augmented reality:
 - A reference anchor must be defined in order for discovery to detect a scene.
 - Annotation anchors must be defined in order to modify language or annotations.

Context

When the developer has annotated a mobile application properly, you can add or delete language translations, edit and delete scenes, and edit annotations from Mobile Services cockpit (with limitations).

Keep in mind the following:

- You can see list and detail views for scenes, edit scenes, and delete them. You cannot create new scenes from Mobile Services cockpit.
- You cannot create or delete annotation anchors from Mobile Services cockpit.
- For a single-app cockpit, you can see the list and detail views for scenes, but cannot edit or delete them from Mobile Services cockpit.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, then select [Mobile Augmented Reality](#) under [Assigned Features](#) (or add it first).
3. Select [Scenes](#) to view a list of scenes associated with the application. The scenes appear in the order they were created. Use filtering and sorting features to find the scenes in which you are interested. To manage the scenes, see [Editing an Augmented Reality Scene](#)

Augmented Reality Scenes

Property	Description
Scene ID	The identifier associated with the augmented reality scene. An anchor reference is required to detect a scene.
Alias	An understandable name associated with the scene.
Creator	The person who created the scene.
Create Time (UTC)	The date and time the scene was created, in YYYY-MM-DD HH:MM:SS:SSS format.
Update Time (UTC)	The date and time the scene was last updated, in YYYY-MM-DD HH:MM:SS:SSS format.
Languages	The languages to which the scene has been translated. You can add or remove languages. Annotation anchors are required.
Actions	The actions to perform such as show detail, edit, or delete the scene.

4. Select [Info](#) to see feature details, such as useful URLs, and to export data in JSON format.


[Editing an Augmented Reality Scene \[page 155\]](#)

Manage augmented reality scenes for the selected application.


1.5.5.1.3.12.1 Editing an Augmented Reality Scene

Manage augmented reality scenes for the selected application.

Prerequisites




The developer creates the application with language codes, scenes, and entities. Language codes must be [ISO 639-1](#)  standardized two-letter codes.

Context

A scene represents a real word with multiple entities. For discovery, a scene must include a `referenceAnchor`. The default is an image, for example a `qrCode`. Entities can be placed relative to the position of the `referenceAnchor`. Such an entity is called an `annotationAnchor`. The position is visually represented by a marker (icon) and associated information is visually represented in the form of a card. To learn more, see the [SAP Fiori for iOS Design Guidelines](#)  for augmented reality (AR) topics. Both iOS and Android apps support augmented reality.

From Mobile Services cockpit, you can modify some of the augmented reality properties and set the languages for the scenes, as described in *Configuring Mobile Augmented Reality*.

Procedure


1. For the selected application, navigate to  [Mobile Augmented Reality](#)  [Scenes](#) .
2. Select a scene from the list. Use the search and sort options to narrow the search to a specific scene or scenes.
3. Under [Language](#), one or more languages are listed for each scene, separated by commas.

Select [+](#) to add one or more languages to the scene.

1. In [Add a language for Scene](#), select a language and then set each annotation anchor for the scene.
2. Click [Finish](#) to save. The language appears in the [Language](#) column.


Select [X](#) to remove a language from the scene. Note that you must keep at least one language.

1. In [Remove language for Scene](#), confirm that the language shown (the ISO-639-1, Native, and English names appear) should be removed for the scene.
 2. Click [OK](#) to save. The language no longer appears in the [Language](#) column.
4. Under [Actions](#), select [Detail](#) to view details about the scene. You can select another language to verify that the annotation anchor is available in a language.
 5. Under [Actions](#), select [Edit](#) to edit the scene. Currently you can edit the [Alias](#) and the [Marker](#) and [Card](#) properties for each of the Annotation Anchors.
 1. In [Edit Scene](#), you can modify the [Alias](#) and select a [Language](#).

2. Under [Annotation Anchors](#) select one of the available annotations to see its Marker and Card details, for example Battery.
3. Under [Marker](#), select icons to represent various functions, such as Play. You can select common icons that can be used for both iOS and Android, or platform-specific icons for iOS and Android. For information, see [AR Markers](#) .

Marker Icons

Property	Description
Icon	Select an icon from a common set supported by both SDKs. For example, a common icon for Play.
iOS Icon	Select a platform-specific icon for iOS (icon-ios).
Android Icon	Select a platform-specific icon for Android (icon-android).

4. Under [Card](#), fill out the Card properties. The Action Type, Action Data, and Action Text together form the action button on the card. For example, a Play Video button might be configured to open a training video. For information, see [AR Cards](#) .

Card Properties

Property	Description
Title	The title for the Card, for example Jump Car Battery.
Description	The title and subtitle for the Card, such as Instructional Video, and Car Batteries.
Action Type	For an Action Button, the Action Type informs the SDK how to handle the action after the user clicks the button. For example, to open a link or video, or to make a call.
Action Text	For an Action Button, the Action Text is the text that appears on the action button, such as Play Video.
Action Data	For an Action Button, the Action Data could be a link URL, video URL, or phone number.
Image	The image associated with the Card.

5. Click [Save](#), or select another Annotation Anchor to modify its properties.
6. Under [Actions](#), select [Delete](#) to remove the scene from the selected application, and select [Yes](#) to confirm.

Related Information

[Configuring Mobile Augmented Reality \[page 153\]](#)

1.5.5.1.3.13 Configuring and Building Apps with the Cloud Build Service

(Not available in all COUNTRY/REGIONS) Use the cloud build service to configure and build clients in Mobile Services cockpit, and enable them to use the SAP BTP SDK.

Prerequisites

- Identify the icons to implement. The file size of the application splash screen and the launch icon can't exceed 1MB.
 - Application launch: 152 by 152 pixels
 - (iOS only) Splash screen: portrait mode (1536 by 2048 pixels)
 - (iOS only) Splash screen: landscape mode (2048 by 1536 pixels)
 - (Android only) Application Icon: application splash page icon
 - (Android only) Branding: corporate logo for branding (displayed only in Android 12 and higher)
 - (Android only) Notification icon: push notification (192 by 192 pixels)

Note

This option only appears if you upload a `google-services.json` file, which is an indication that you plan to use Android notification.

Note

Supported image formats:

- For iOS: PNG and JPG.
 - For Android: PNG, JPG, and Vector XML (Android 12 and higher for the latter).
- Create signing profiles needed for each platform type. See *Manage Signing Profiles*.
 - **(Mobile Development Kit client only):** requires App Update feature. Please upload the bundle file to App Update feature, and deploy.

Context

Cloud build service in Mobile Services cockpit supports the following:

- Mobile Development Kit client
- Customized Mobile Development Kit client
- SAP Asset Manager apps

For information about the different client types and required packaging details, see *Supported Client Types*, *Packaging Details*, and *Build Options*.

Note

The apps are automatically added to App Lab, which makes it easy for developers and testers to install and use during the development cycle, and for administrators to store the binaries.

The App Lab process that runs in the background sends an event log at the INFO level when everything works fine, or at the ERROR level when processing fails. If you don't see an app you expect, or want more information, navigate to ► [Analytics](#) ► [Logs](#) ► and view [Event Logs](#).

You can also rebuild an existing binary using different build options.

After building the app, you can download the app binary and build logs or enable over-the-air (OTA) installation for the binary.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select an application, then select [Mobile Cloud Build](#) under [Assigned Features](#) (or add it first).
3. On [Build Jobs](#), view a list of available build jobs for the selected application.

Build Job Properties

Property	Description
ID	The generated build job identifier. This ID can be useful for searching the logs for information in a troubleshooting situation.
Build Job Name	The name assigned to the build job, such as "Test App".
Client Type	The client type used for the build job, such as Mobile Development Kit client or SAP Asset Manager. (Not available for build jobs created prior to the November 2021 release).
Status	The current status of the build job, such as "Build Completed" or "Building" or "Build Failed".
Signing Profile	The signing profiles used for the platforms included in the build job, in the format <code><Platform>:<signing_profile_name></code> . For example: <code>Android:AndroidProfileKey0</code> or <code>iOS:CBSComSapFioriWildcard</code> .
Version	The version assigned to the build, such as 1.0.
Actions	The actions you can perform, such as build, install, edit, and delete.

- [Create Build Job](#) button – create a new build job. See [Creating a Build Job](#).
- Select [Build](#) to run the selected build job. Check the [Status](#) column to view progress.
- Select [Install](#) to obtain the link to install an app on a device, if a binary is available. This option is available for the `APK` and `IPA` build format. You can do one of the following:
 - Scan the QR code with a device.

- Copy the URL to the clipboard. To install the binary, paste and access the URL.
 - [Edit](#) – edit the selected build job. See *Editing a Build Job*.
 - [Delete](#) – delete the selected build job.
4. You can select each build job to view its current state. If the binary format is **APK** and **IPA**, an [Install](#) section appears on the page.

Cloud Build History

Item	Description
Build State	Current state of the build, such as Building, Build Failed, and Build Complete.
Project Generation State	(Mobile Development Kit and Custom Mobile Development Kit only) If you selected Save Project as a build option during configuration, this column appears and provides the current state of the project build. States include Generating, Generation Complete, Generation Failed and N/A.
Debug Enabled	Whether the debugging tool is enabled.
Operating System	The operating system of the build, such as iOS or Android.
Binary Format	The binary format used for packaging, either (AAB) Android App Build or APK (Android Packaging Kit) if the platform is Android; IPA (iOS App Store Package) if the platform is iOS.
Form Factor	The device type, such as tablet or phone.
Version / Version Name	The application version.
Build / Version Code	The build version.
SDK Version	The SDK version used for the build job, such as 1.1 or 2.3.0. (Not available for build job history created prior to the November 2021 release).
Built On	The date of the build, in the format YYYY-MM-DD HH:MM (local or UTC).
Certificate Expiration	The date the certificate expires, in the format YYYY-MM-DD HH:MM (local or UTC).
Actions	Actions you can take, if any, such as Install or Download Binary.

- Select [Install](#) to install the binary (APK and IPA), either by scanning the QR code, or copying the URL.
- Select [Download Binary](#) to download the binary to a local directory. When ready, you can post the binary to a download site.

- Select [Download Project](#) to download a project to a local directory. The project contains all artifacts used to build the project.

5. (Optional) Select the build job history row to drill down further to see additional build details.

- Under [Install](#), you can install the binary (APK and IPA), either by scanning the QR code or copying the URL.
- Under [Build Logs](#), you can view the live build logs as the build progresses. The log includes a summary of build attributes appropriate for the platform, such as the client type, tool set versions, plugins, Cordova versions, and SAP UI5:
 - iOS – platform, application version number, type, Kapsel version, node information, Cordova and Xcode versions, and the selected operating system.
 - Android – platform, type, application version number, Kapsel version, node information, Cordova versions, node_js, gradle, Java, and SDK tool versions, and the selected operating system.
 - SAP UI5 – applies to iOS and Android builds for packaged apps .

You can select [↓](#) and download the logs to a local directory, if required. Use the logs to troubleshoot problems.

The following options are also available:

- Select the [Download Binary](#) button to download the binary.
- (Mobile Development Kit and Custom Mobile Development Kit only) Select [Download Project](#) to download the complete project source code that is generated by the build for debugging the app. Choose this option to download a ZIP file that contains the project structure of the specified application and its related UI5 libraries, plugins, device platform, resource details, generated files, and any dependencies that are related to the iOS or Android platforms. To use this option, make sure that you choose [Save Project](#) as a build option before you initiate the build. Also, ensure Xcode is installed for iOS apps, and Android Studio IDE is installed for Android apps. After downloading the project, validate it by re-building the project using Android Studio IDE.

Note

Use the existing version of Android Studio IDE. Do not update the Android Gradle plugin, when you are prompted.

1. Unzip the downloaded project zip file.
 2. Launch Android Studio IDE and choose [File > Open](#) to open the Android project. It may take a few minutes to create the file index initially.
 3. Choose [Build > Rebuild Project](#).
The build must complete successfully.
- Select the [Delete](#) button to delete the binaries. Any binary that was added to App Lab will automatically be deleted as well.

6. Select the [Info](#) tab to view feature details.

Supported Build Types, Client Types, Packaging Details, and Build Options [page 161]

The cloud build service supports different build types, client types, packaging details, and build options.

Creating a Build Job [page 168]

Create a build job for the selected application.

Creating a Build Job that Uses Certificates [page 174]

You can use the Cloud Build service to build a Mobile Development Kit or SAP Asset Manager client with certificate-based authentication. When enabled, the client can access a certificate on the device during onboarding, if requested by the identity provider (IdP).

[Editing a Build Job \[page 175\]](#)

Edit a build job for the selected application.

Related Information

[Supported Build Types, Client Types, Packaging Details, and Build Options \[page 161\]](#)

[Troubleshooting: Common Issues \[page 318\]](#)

[Supported Build Types, Client Types, Packaging Details, and Build Options \[page 161\]](#)

[Managing Cloud Build Service Settings \[page 277\]](#)

[Create a Signing Profile \[page 277\]](#)

[Purge Cloud Build Artifacts \[page 279\]](#)

[Supported Build Types, Client Types, Packaging Details, and Build Options \[page 161\]](#)

1.5.5.1.3.13.1 Supported Build Types, Client Types, Packaging Details, and Build Options

The cloud build service supports different build types, client types, packaging details, and build options.

Supported Build Types

Build Type	SDK Version	UI5 Versions	Supported Android Versions	Supported iOS Versions	Notes
Mobile Development Kit (MDK)	23.12.1 (default version), 23.8.7, and 23.4.6	N/A	MDK 23.12.1, 23.8.7, and 23.4.6: 14.0, 13.0, 12.0, 11.0, 10.0, 9.0, 8.1, 8.0	MDK 23.12.1: 17.4, 17.0, 16.7, 16.0, 15.7, 15.0, 14.8, 14.0 MDK 23.8.7 and 23.4.6: 17.2, 17.0, 16.7, 16.0, 15.7, 15.0, 14.8, 14.0	MDK 23.12.1 and 23.8.7 use NativeScript 8.6.3 and Node 16.14.2 MDK 23.4.6 uses NativeScript 8.4.0 and Node 16.14.2
Customized Mobile Development Kit					
SAP Asset Manager (SAM)	2310.0.2 (default version), 2305.0.2, 2210.0.7, and 2205.0.3	N/A	SAM 2310.0.2, 2305.0.2, 2210.0.7, and 2205.0.3: 14.0, 13.0, 12.0, 11.0	SAM 2310.0.2, 2305.0.2, 2210.0.7, and 2205.0.3: 17.2, 17.0, 16.7, 16.0, 15.7, 15.0	SAM 2310.0.2, 2305.0.2, 2210.0.7 and 2205.0.3 use:

Build Type	SDK Version	UI5 Versions	Supported Android Versions	Supported iOS Versions	Notes
					<ul style="list-style-type: none"> • Mobile Development Kit 23.8.4 • SAM Plugin 2310.0.200 • NativeScript 8.6.1 • Node 16.14.2

Note for Customized Mobile Development Kit build:

Note

For application error due to `Used by both Neo Could not find module 'tns-core-modules/file-system'` or similar messages, refer to [Metadata / Client Project Migration](#) for application metadata migration.


Supported Client Types

Client Type	Required Packaging Details See Packaging Details Overview [page 164]	Supported Build Options See Build Options Overview [page 168]
Mobile Development Kit client See SAP Mobile Service, Mobile Development Kit .	<ul style="list-style-type: none"> Build Version Build Job Device App Name Device App Display Name Device App Version Device App Details Bundle ID (iOS) / Package Name (Android) Secure Database Encryption Allow Certificate Client ID (OAuth only) Redirect URL (OAuth only) <p>Android and iOS Builds:</p> <p>URL Scheme</p> <p>Android Builds:</p> <ul style="list-style-type: none"> Google Services JSON File Binary Format <p>Multimedia:</p> <p>Application Launch icon</p>	<ul style="list-style-type: none"> Create Debug-enabled Binaries Signing Profile SDK Version Minimum Platform Version Save Project
Customized Mobile Development Kit client	<ul style="list-style-type: none"> Build Job Name Secure Database Encryption MDK Project ZIP File <p>Android Builds:</p> <ul style="list-style-type: none"> Google Services JSON File Binary Format <p>Multimedia:</p> <p>Application Launch icon</p>	<ul style="list-style-type: none"> Create Debug-enabled Binaries Signing Profile SDK Version Minimum Platform Version Save Project

Client Type	Required Packaging Details	Supported Build Options
	See Packaging Details Overview [page 164]	See Build Options Overview [page 168]
SAP Asset Manager See SAP Asset Manager .	<ul style="list-style-type: none"> • Build Version • Device App Name • Device App Display Name • Device App Version • Device App Details • Bundle ID (iOS) • Package Name (Android) • Secure Database Encryption • Allow Certificate • Client ID (OAuth only) • Redirect URL (OAuth only) <p>Android & iOS Builds:</p> <p>URL Scheme</p> <p>Android Builds:</p> <ul style="list-style-type: none"> • Google Services JSON File • Binary Format <p>Multimedia:</p> <p>Application Launch icon</p>	<ul style="list-style-type: none"> • Create Debug-enabled Binaries • Signing Profile • SDK Version • Minimum Platform Version

Packaging Details Overview

Field	Description
Build Job Name	(Customized Mobile Development Kit client only) A unique build job name to replace the default name.
MDK Project ZIP File	(Customized Mobile Development Kit client only) The customized Mobile Development Kit project file to upload. The file must be in ZIP format, and meet validation rules.

Field	Description
Device App Name	<p>A user-friendly name for the device application that appears as label on the device, such as Fiori Weather App. The Device App Name is not the same as the App Name or the application ID property you configured when you defined the app, and becomes the default for this field.</p> <div>  Note "Cordova" is a reserved word, so you cannot use it in the Device App Name. </div>
Device App Version	The application version.
Device App Display Name	The label that appears on the welcome screen after you launch the app. This must be a String value with a maximum length of 128 characters. See the note below when building client type Mobile Development Kit Client with MDK 6.2 and later.
Device App Details	The subtitle for the application, which appears on the app's welcome page. This must be a String value with a maximum length of 128 characters.
Bundle ID (iOS)	<p>The unique app identifier provided by the App Store.</p> <p>You can change the bundle ID to match the bundle ID of the provisioning profile that you uploaded while creating the signing profile used to sign the iOS Mobile Development Kit and SAP Asset Manager clients.</p>
Package Name (Android)	The unique app identifier. For Android apps, the package name used to sign Android Mobile Development Kit clients.
URL Scheme	<p>For iOS builds: The custom URL scheme that enables other apps to communicate with your app.</p> <p>For MDK, use the URL Scheme entry for both iOS and Android builds.</p> <p>For Android builds: A related webpage URL for any card template, that enables the user to open the relevant webpage for each card instance.</p> <p>For MDK, use the URL Scheme entry for Android builds, not Deep Link URL.</p>

Field	Description
Google Services JSON File	<p>For Android builds: The Firebase Android configuration file associated with your app in your Firebase project. Use Browse to navigate to and select this file.</p> <p>The file is uploaded to the service when you save the Cloud Build configuration. To display the contents of the JSON file after upload, select Show Contents.</p> <p>To remove the uploaded Google Services JSON File, select Remove File.</p>
Binary Format	<p>For Android builds: The packaging format to use for the build, including APK (Android Package Kit, the default) or AAB (Android App Bundle). Since Google requires applications uploaded to the Google Play Store be built in the AAB format, select this option if that is your plan. To install an AAB binary without using Google Play Store, you must download the AAB and use Google's <code>bundletool</code> to extract an install-ready binary from the AAB and to install that binary. Refer to their documentation on <code>bundletool</code> for more details.</p> <p>For iOS builds: The packaging format to use is IPA (iOS App Store Package).</p>
Certificate Authentication for OAuth	<p>Set OAuth to use certificate-based authentication.</p> <p>For Android builds: Certificate-based authentication that uses the SAP Mobile Cards client's user certificate to satisfy the OAuth challenge during onboarding. The user does not have to enter their credentials in the identity provider (IDP) screen.</p>
Secure Database Encryption	<p>Whether the database must be encrypted. True indicates the database should be encrypted. False indicates the database should be encrypted with a known key MDK (MDK ≥ 6.2) or unencrypted (MDK < 6.2). Use the False option to debug the database content (this is meant to be used for development, debugging or demonstration purposes only).</p>

Field	Description
Allow Certificate	<p>Whether to allow Mobile Development Kit and SAP Asset Management clients to request a certificate from the client device during onboarding for certificate-based authentication. The client type must implement the OAuth security configuration, and you must configure the Redirect URL (it cannot use the default). Both are configured in the application security settings.</p> <p>When Allow Certificate is checked and the selected redirect URL starts with <code>http://</code> or is enabled, an error appears for Client ID informing you that the Redirect URL is not valid because Allow Certificate is selected.</p> <p>When Allow Certificate is checked, the selected Client ID is valid, and URL Scheme (Platform page) is empty, Redirect URL is populated automatically with the URL Scheme prefix. Alternatively, you can input the value manually and save.</p> <p>When Allow Certificate is checked, the selected Client ID is valid, but the URL Scheme value (Platform page) doesn't match the Redirect URL prefix, an error appears for URL Scheme informing you that the URL Scheme must match the prefix of the redirect URL because Allow Certificate is selected.</p>
Client ID (OAuth only)	<p>(Mobile Development Kit and SAP Asset Management only) Select an existing client ID option or create a specific client ID for a client build configuration. The default configuration is "Always Use Default OAuth Client ID". This option follows the original behavior of triggering the build, which uses the default OAuth client to trigger build.</p> <p>You can create multiple OAuth client configurations in application security (the first configuration in the list becomes the default). This enables you to select a specific client ID for a client build option. You can use a temporary client ID while configuring a cloud build configuration, but must configure a valid client ID before running the cloud build job.</p>
Redirect URL (OAuth only)	<p>(Mobile Development Kit and SAP Asset Management only) Either select the automatically generated universal link or enter a valid redirect URL. For example, <code>RedirectURL="myurlscheme://oauth2redirect"</code>.</p>

Note

When building client type Mobile Development Kit Client with MDK 6.2 and later:

- If a value is provided in the [Device App Display Name](#) field, the same value is also used as the launcher name, which is the text under the application icon on the device screen. When specifying this value, please consider the launcher naming recommendations made by various app stores, and possible consequences if they are not followed.
- Otherwise, the launcher name uses the value of the [Device App Name](#) field.

Build Options Overview

Build Option	Description
Create Debug-enabled Binaries	Whether to build the binary with debug enabled.
Signing Profile	The signing profile with which to sign the binary.
SDK Version	Select the SDK version to use for the build, or use the SAP recommended version.
Minimum Platform Version	The minimum device operating system version required to install the binary.
Save Project	Whether to save the project so you can download it later using Download Project , if required.

Related Information

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

[Troubleshooting: Common Issues \[page 318\]](#)

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

1.5.5.1.3.13.2 Creating a Build Job

Create a build job for the selected application.

Context

A build job merges both project settings, and build settings, such as signing profiles. Select the [Create Build Job](#) button to create a build job for Mobile Development Kit Client, or SAP Asset Manager.

As a developer you can configure multiple Custom Mobile Development Kit Client configurations in parallel to experiment with branding and other build settings. You can easily switch between configurations, see the build

job status, drill down to the build job details and executions logs, and experiment with installing the app on the device via QR Code. See *Creating a Build Job for Customized MDK Clients* for details.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select an application, then select [Mobile Cloud Build](#) under [Assigned Features](#) (or add it first).
3. Select [Create Build Job](#) to create a new build job.
4. In [Create Build Job](#), select a [Client Type](#) from the list.

Supported client types include:

- Mobile Development Kit Client
- Customized Mobile Development Kit Client (see *Creating a Build Job for Customized MDK Clients*)
- SAP Asset Manager

For information about the different app types and required packaging details, see *Supported Client Types, Packaging Details, and Build Options*.

5. On the [Basic Information](#) page, fill out the information for the selected client type. The fields are dynamic and appear depending on the selections you make.

For information about additional entries see *Supported Client Types, Packaging Details, and Build Options*.

6. On the [Platform](#) page, fill out the platform-related information page.
 - (Mobile Development Kit Client) Under [Android & iOS Builds](#), enter the URL Scheme. Under [Android Builds](#), enter the [Google Services JSON File](#), or browse to find it. Select [Show Contents](#) to view the contents of the JSON file. If it is not the right file, click [Remove File](#) and try again. Once saved, the JSON file is uploaded. Select the binary format to use, either APK (Android Package Kit, the default) or AAB (Android App Bundle).
 - (SAP Asset Manager) Under [Android & iOS Builds](#), enter the URL Scheme. Under [Android Builds](#), enter the [Google Services JSON File](#), or browse to find it. Select [Show Contents](#) to view the contents of the JSON file. If it is not the right file, click [Remove File](#) and try again. Once saved, the JSON file is uploaded. Select the binary format to use, either APK (Android Package Kit, the default) or AAB (Android App Bundle).
7. On the [Multimedia](#) page, identify the launch and splash screen icons, depending on the selected app type. You can use the default icon, or [Upload](#) a new icon following the size guidelines, or click [Reset](#) to revert to the default. Click the image title to preview the icon.

For Android builds, you can customize the notification icon.

8. On the [Build Options](#) page, provide the build options:
 - Platforms – the platforms to build, such as Android and iOS. Mobile Development Kit Client, and SAP Asset Manager apps support Android and iOS platforms.
 - Signing Profile – the profiles for the Android and iOS platforms. To build an iOS Mobile Card Kit client containing iMessage and Today Screen extensions, choose individual signing profiles for each extension.

Note

If an older signing profile reports a Null Object Exception, you must reload the profile.

- Minimum Platform Version – the minimum operating system version required for the Android and iOS platforms.
- SDK Version – select a supported version from the list. The default version is always recommended (it will vary over time), but you can select an older SDK version. The message *The SDK version is no longer supported* indicates you should build with a newer version.
- Build Options – define the type of binary to build.
 - Create Debug-enabled Binaries
 - Save Project

For information about the different build options, see *Supported Client Types, Packaging Details, and Build Options*.

9. Select *Finish*.

[Creating a Build Job for Customized MDK Clients \[page 170\]](#)

Create a build job for a customized Mobile Development Kit client from Mobile Services cockpit. The build includes functionality to run customized extensions, application resources, and onboarding; and to run demo mode in the Cloud Foundry landscape.

1.5.5.1.3.13.2.1 Creating a Build Job for Customized MDK Clients

Create a build job for a customized Mobile Development Kit client from Mobile Services cockpit. The build includes functionality to run customized extensions, application resources, and onboarding; and to run demo mode in the Cloud Foundry landscape.

Prerequisites

The custom Mobile Development Kit project ZIP file must be prepared in a valid format, and must be valid for the current MDK version available in the Cloud Build service, otherwise you may see build failures, or event failure while running the app. Following is an example of the core structure that follows validation rules, but is not an exhaustive representation of the supported content:

```
template.mdkproject.zip
├── xxxxx.mdkproject/
│   ├── App_Resources/
│   │   ├── Android/
│   │   │   ├── google-services.json
│   │   │   └── src/
│   │   │       ├── main/
│   │   │       └── assets/
│   └── iOS/
├── BrandedSettings.json (Required)
├── MDKProject.json (Required)
├── demo/
├── extensions/
└── metadata/
```

The Cloud Build service expects the uploaded file to contain one root folder (which includes the `.mdkproject` suffix). To do this, the developer must compress the local `.mdkproject`.

Context

When the MDK project ZIP file is uploaded, the following validation is performed on the file:

- The file must pass a virus scan, and report an error if it fails.
- The file must be in a ZIP file format, and report an error if not.
- The ZIP file content is processed iteratively to verify the following:
 - Your `xxxxxx.mdkproject` is at the root of the ZIP file.
 - `BrandedSettings.json` is directly under the root folder `xxxxxx.mdkproject`, and report an error (bad request) if not.
 - `MDKProject.json` is directly under the root folder `xxxxxx.mdkproject`, and report an error (bad request) if not.
 - `BrandedSettings.json` and `MDKProject.json` can be parsed, and report errors (internal server error) if not well-formatted.
- The above-parsed values are validated using the following rules:
 - `AppName` in `MDKProject.json` (required): only letters and digits are allowed in the name, and the name can be no longer than 80 characters.
 - `AppVersion` in `MDKProject.json` (required): must match one of the regular expressions:

```
/^[1-9][0-9]{0,3}.\([0]|[1-9][0-9]{0,1})$/  
/^[1-9][0-9]{0,3}.\([0]|[1-9][0-9]{0,1})\.\([0]|[1-9][0-9]{0,1})$/
```

- Must include either two or three segments, separated by periods (".")
 - Each segment contains only digits.
 - The first segment must start with a digit greater than 0, followed by 0 or 3 digits.
 - The second and third segments must include a number from 0 to 99.
- `UrlScheme` in `MDKProject.json` (optional): must start with a letter, followed by letters, digits, plus sign ("+"), period ("."), or dash ("-"), and cannot be longer than 255 characters.
- `BundleID` in `MDKProject.json` (required): must start with a letter, followed by letters, digits, period ("."), or underscore ("_"), and can be no longer than 64 characters.
- `Google-services.json` (optional): must be a valid JSON file, and `BundleID` in `MDKProject.json` must be one of the packages listed in `google-services.json`.
- `ApplicationDisplayName` in `BrandedSettings.json` (optional): the length is checked, and can be no longer than 128 characters.
- `DetailLabelViewText` in `BrandedSettings.json` (optional): the length is checked, and can be no longer than 128 characters.

Procedure

1. Select [Create Build Job](#) to create a new build job.
2. In [Create Build Job](#), for [Client Type](#) select [Customized Mobile Development Kit Client](#) from the list.
3. On the [Basic Information](#) page, fill out the information for the customized Mobile Development Kit Client.
 - For [Build Job Name](#), enter a name to replace the default name.
 - For [MDK Project ZIP File](#), provide the name of the project file to use, or browse to select it. You can also upload a new version of the project file, for example, after fixing a validation failure during upload.

If the project file cannot be uploaded, check for error reports, and make changes until all validation checks can be completed successfully.

If the project file is valid, [Device App Name](#) and the following fields are updated based on values contained in the ZIP file. In some cases, you may need to modify connection properties (see [Connection Properties](#) for information).

Select [Next](#) to navigate to subsequent pages. For information about entries see [Supported Client Types](#), [Packaging Details](#), and [Build Options](#). Some values can be modified, others are read-only.

4. On the [Platform](#) page, fill out the platform information page.
 - Under [Android & iOS Builds](#), enter the URL Scheme.
 - Under [Android Builds](#), you can browse to check the existing `google-services.json` file content contained in the ZIP file, or you can upload a new `google-services.json` file to override it. Also select the binary format to use, either APK (Android Package Kit, the default) or AAB (Android App Bundle).
Once saved, the `google-services.json` file is uploaded. Once uploaded, select [Show Contents](#) to view the contents of the JSON file. If it is not the right file, click [Remove File](#) and try again.

Select [Next](#) to navigate to subsequent pages. For information about entries see [Supported Client Types](#), [Packaging Details](#), and [Build Options](#). Some values can be modified, others are read-only.

5. On the [Multimedia](#) page, identify the launch and splash screen icons, depending on the selected app type. For Android builds, you can customize the notification icon. You can use the default icon; [Upload](#) a new icon to override what is contained in the ZIP file, following the size guidelines; or click [Reset](#) to revert to the default. Click the image title to preview the icon.

Select [Next](#) to navigate to the subsequent page. For information about entries see [Supported Client Types](#), [Packaging Details](#), and [Build Options](#). Some values can be modified, others are read-only.

6. On the [Build Options](#) page, you can modify the build options for the customized Mobile Development Kit Client.
 1. [Signing Profile](#) – select signing profiles for the Android and iOS platforms.

Note

If an older signing profile reports a Null Object Exception, you must reload the profile.

2. [Minimum Platform Version](#) – select the minimum operating system version required for the Android and iOS platforms.
3. [SDK Version](#) – select a supported version from the list. The default version is always recommended (it will vary over time), but you can select an older SDK version. The message `The SDK version is no longer supported` indicates you should build with a newer version.

4. [Create Debug-enabled Binaries](#) – create a binary version with debug tools enabled.

For information about build options, see *Supported Client Types, Packaging Details, and Build Options*.

7. Select [Finish](#). If all entries are valid, the new job appears in the build job list.

Next Steps

Check the [Build Jobs](#) tab to see the status of your build. Status values include:

- [Build Required](#) – the build job has been created. Select [Build](#) under [Actions](#) to run the build job.
- [Build Completed](#) – the build job ran successfully.
- [Build Failed](#) – the build failed. Investigate the cause, and then run the build job again.

1.5.5.1.3.13.2.1.1 Connection Properties

Connection properties may be set automatically by the build job or you may need to set them manually depending on several conditions.

The `AppId` in the `ConnectionSettings` section of the `BrandedSettings.json` file is always set by the build job using the ID value of the application where the build job is created. Other `ConnectionSettings` properties (`ClientId`, `ServerUrl` (and / or `SapCloudPlatformEndpoint`, see note below), `AuthorizationEndpointUrl`, `RedirectUrl`, and `TokenUrl`) are also set by the build job if the property is not present or its value is set to empty (") in the `BrandedSettings.json` file. Use this approach when an app is going to onboard to the same landscape where the app binary was built.

Note

Starting with Mobile Development Kit version 5.2, `SapCloudPlatformEndpoint` is renamed to `ServerUrl`. For backward compatibility, both properties are supported by Mobile Development Kit. If both properties are defined in the `BrandedSettings.json` file, then Mobile Development Kit uses `ServerUrl`. Merge rules include:

- Both properties are set to the same value in the end.
- If both properties are missing or empty or start with a placeholder, then the value is set by the build job.
- If `SapCloudPlatformEndpoint` has a value but `ServerUrl` is missing or empty or starts with a placeholder, then the `SapCloudPlatformEndpoint` value is used.
- If `SapCloudPlatformEndpoint` is missing or empty or starts with a placeholder but `ServerUrl` has a value, then the `ServerUrl` value is used.
- If both `SapCloudPlatformEndpoint` and `ServerUrl` have values, then the `ServerUrl` value is used.

If an app is going to onboard to a different landscape from where the app binary is built, you must set each connection property in `BrandedSettings.json` with the corresponding property value of the same application in the onboarding landscape.

For the app to onboard successfully, the application ID must be the same in both the build and onboarding landscapes.

Another condition under which the properties mentioned above are set by the build job, is if the value starts with "<Enter" after stripping spaces from both start and end. The condition is added because if `BrandedSettings.json` is copied from the default template `mdkproject`, the mentioned properties usually come with placeholder values. The values are not empty but they are not correct, so the build job deems they need to be populated with correct values.

1.5.5.1.3.13.3 Creating a Build Job that Uses Certificates

You can use the Cloud Build service to build a Mobile Development Kit or SAP Asset Manager client with certificate-based authentication. When enabled, the client can access a certificate on the device during onboarding, if requested by the identity provider (IdP).

Prerequisites

The client type must implement the OAuth security configuration, and you must configure the Redirect URL (it cannot use the default).

Context

See *Creating a Build Job* for detailed instructions. These steps focus on special instructions for building the Mobile Development Kit or SAP Asset Manager client with certificate-based authentication.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, then select **Mobile Cloud Build** under **Assigned Features** (or add it first).
3. Select **Create Build Job** to create a new build job.
4. In **Create Build Job**, select a **Client Type** from the list.

Supported client types for the certificate option include:

- Mobile Development Kit Client
- Custom Mobile Development Kit Client also supports the certificate option, but is read-only. The **Allow Certificate** value is based on the uploaded project ZIP file.
- SAP Asset Manager

For information about the different app types and required packaging details, see *Supported Client Types*, *Packaging Details*, and *Build Options*.

5. On the **Basic Information** page, fill out the information for the selected client type. The fields are dynamic and appear depending on the selections you make.

Select [Allow Certificate](#) to enable the Mobile Development Kit or SAP Asset Manager client to access a certificate on the client device during onboarding if it is requested by the identity provider (IdP).

For information about additional entries see *Supported Client Types*, *Packaging Details*, and *Build Options*.

6. Complete entries for the rest of the wizard pages.

Note


On the [Platform](#) page, keep in mind that for [URL Scheme](#), make sure that the [URL Scheme](#) uses the same value as the [Redirect URL](#). For example, if the [Redirect URL](#) is `RedirectURL="myurlscheme://oauth2redirect"`, then the [URL Scheme](#) should be `"myurlscheme"`, or else the Mobile Development Kit client may fail to onboard.

7. Select [Finish](#).

1.5.5.1.3.13.4 Editing a Build Job

Edit a build job for the selected application.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, then select [Mobile Cloud Build](#) under [Assigned Features](#) (or add it first).
3. On the [Build Jobs](#) tab, under [Actions](#) select  to edit a build job.
4. Use the [Edit Build Job](#) wizard to make changes to the build job. See *Creating a Build Job* for field definitions. The fields that appear on each page depend on the client type selected.
5. Select [Finish](#). The build job [Status](#) changes to "Build Required", and the new values are used the next time you run the build.

1.5.5.1.3.14 Managing Application Versions Using App Lab

Developers can manage internal application versions during the development and test cycle using the App Lab service.

Prerequisites

If you upload application files containing Unicode characters, use a Firefox or Chrome browser, instead of an Internet Explorer browser. You may encounter upload problems if you use Internet Explorer.

Context

Instead of storing apps on individual developer devices and using email to distribute apps for testing and review, you can manage internal versions for each application and make them available via the App Lab service. Upload multiple application versions for Android and iOS, record notes and instructions, and make apps public to your internal audience.

Very large iOS or Android applications are uploaded in 5MB chunks, enabling you to resume processing if the upload is disrupted. The progress bar keeps you informed of the status, and a message similar to `uploading Application version file xxxxx.xx . . .` appears while large chunks are uploaded.

To enable access to app versions, provide the App Lab URL to internal users and testers. Your internal users can then download one or more app versions to test and troubleshoot.

Note

If you're a Cloud Platform Cloud Foundry space member administrator, your space member roles also apply when you access Mobile Services cockpit. This enables you to apply role management as expected for all application types (Native/MDK, SAP Mobile Cards, and App Lab), for example:

- Space Developer role – read and write access to the [App Lab](#) page (for example, you can initialize).
- Space Auditor role – read access to the [App Lab](#) page.
- Space Manager role – read access to the [App Lab](#) page.

By default, business users who don't have access to App Lab can't log in, and see the message "You are not authorized for access." As the administrator you, can unselect [Restrict App Lab to authorized users](#) to enable read access for these users.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, then select [Mobile App Catalog](#) under [Assigned Features](#) (or add it first)
3. Select [Artifacts](#).
4. Under [Mobile Application Artifacts](#), for [Application Versions](#), you can do the following:
 - Add a new application version:
 1. Click [+](#).
 2. Browse to an .ipa or .apk application file and select [Upload](#).
After the file is successfully uploaded, [Add Application Version](#) displays read-only information that's been extracted from the file.

Add Application Version Details








Application Version Properties	Description
Application ID	Unique application identifier.

Application Version Properties	Description
Application Name	Application name.
Bundle ID	The resource bundle identifier.
Version	The version assigned to the application.
Platform	Android or iOS.
Build Version	The version number of the build.
Origin	Where the application version was generated. For versions uploaded from cloud build service, CBS is specified.
Form Factor	The device type
Uploaded By	The user who uploaded the app.
Notes	Useful notes to help the tester.
Publish to App Lab	Makes the app available for internal users to install the application for testing.
Contact Info	Contact information, in case users need to contact you with questions or feedback. For example, @jj_support or devops@mycompany.com .
Application	Name of the .ipa or .apk application file that you uploaded. In edit mode, select Download to download the application binary from within Mobile Services cockpit. The Download button appears only in edit mode.

3. Provide any additional information as required and click [Save](#).


- View application versions that have already been uploaded.

5. After an application has been uploaded successfully, you can perform the following actions:

- Edit the version to change the contact information or version notes by choosing .
- Download the selected version by choosing .
- Toggle the published state of the version by choosing  to publish or  to unpublish.
- Upload a new version by choosing .
- Install the version by choosing .
- Delete the version by choosing .

6. Publish the app version to the App Lab service. The App Lab service deploys the app version into your Cloud Foundry space. Once the app's been deployed, you can notify users that it's available for testing.

To publish the app version to App Lab:

1. In Mobile Services cockpit, navigate to  [Settings](#)  [App Catalog](#)  [App Lab](#) .

Note

Remember, if you're a Cloud Platform Cloud Foundry space member administrator, your space member roles also apply when you access ► [Settings](#) ► [App Catalog](#) ► [App Lab](#) ▾ from Mobile Services cockpit, as described above.

As an administrator, you can indicate whether users without the "Viewer" scope have read access to App Lab. By default, [Restrict App Lab to authorized users](#) is selected, which prevents users without the "Viewer" scope from logging in to App Lab. If you unselect the option, users who aren't assigned the "Viewer" scope can log in with read access to App Lab.

2. On [App Lab](#), click [Initialize](#). You only need to do this once for the Cloud Foundry space you are administering. It may take a moment for the app to become available at the address shown on the page; you may see a 404 Not Found message during this time.
3. Copy the link that appears on the page, and send it to your testers.
7. If you see an [Upgrade](#) button in the upper-right corner, it indicates that App Lab needs to be updated. You may also see the message `Upgrade is required for App Lab. Click Upgrade.`

Select [Upgrade](#) to proceed. The upgrade takes about five minutes.

Next Steps

Once your internal users and testers receive the link, they can use it to access available applications and choose the application version to install. The list of available applications is appropriate for the device; for example, if a tester accesses the link from an iOS device, only iOS application versions appear.

Note

If users attempt to access the address while the app is still being created, they'll see the 404 Not Found. Instruct them to wait a moment before retrying the address.

1.5.5.1.3.15 Exporting Data

SAP Mobile Services allows you to export business configuration data for an assigned mobile service that is bound to an application.

Context

The data is downloaded in a .json file.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select an application.
3. Under [Assigned Features](#), select one of the features assigned to the application, such as [Mobile Settings Exchange](#).
4. Select the [Info](#) tab.
5. In [Data Export](#), select [Export Data](#).
6. Open the .zip file, or save it locally.

The .json includes exported data for the assigned features, in a format similar to:

```
{
  "storageApplicationConfiguration" : [ {
    "appid" : "d055066.test",
    "content" : "{enc}"
    <data_content>,
    "lastupdated" : "2019-05-14T13:06:35Z",
    "tenantName" : "<tenant_name>"
  } ],
  "storageUserConfiguration" : [ ],
  "storageDeviceConfiguration" : [ ]
}
```

1.5.5.2 Enabling Applications to Discover Configurations

(Not available in all COUNTRY/REGIONS) Using Mobile Services cockpit, you can add the application configurations or link the domain from the app itself. You can update or delete the configurations at any time.

Context

For developers who use Discovery Service for app configurations, keep in mind:

- When you create a domain-based configuration from any space in any region, and a client running the app requests the configuration from the global Discovery Service address, the app gets its configuration from your space.
- When you create an onboarding-code-based configuration from any space in any region, and a client running the app requests the configuration from the global Discovery Service address, the app gets its configuration from your space.
- For regions where the global Discovery Service is not available, you can overwrite the Discovery Service URL with a regional one by configuring `STAND_ALONE_DISCOVERY_ADMIN` on SAP BTP. When configured, the server deploys the regional Discovery Hub instead of the global Discovery Hub. In installations where the regional solution is implemented, the regional URL appears on the Mobile Services cockpit ► [Settings](#) ► [Discovery Service](#) ► page for reference.

When customers terminate data service, all configuration items registered for the customer space are deleted.

[Adding Service Provider Domains \[page 180\]](#)

Add a domain for the application service provider.

[Adding Onboarding Codes \[page 181\]](#)

Onboarding codes can be added to the application as an alternative to Service provider domains.

[Adding Application Configurations \[page 182\]](#)

Application configurations can be added at the application level.

1.5.5.2.1 Adding Service Provider Domains

Add a domain for the application service provider.

Procedure

1. **Note**

- Easy communication with the user as there is a registered email domain.
- Increased dependency on other individuals or processes to setup and use service provider domains.

In Mobile Services cockpit, select **Settings** > **Discovery Service**.

2. Click the plus sign.
3. Enter the *Domain*.

Note

To register an email domain, use a sub-domain of a registered email domain, or use one that the Web host service administrator has defined in the DNS TXT record for `_sap_mobsec.<emaildomain>`.

If you want to use your own domain, you must register it first. To do so you must note the `<customerid>` value and create a TXT record with the DNS provider of the domain with key `_sap_mobsec.<emaildomain>` and value for the `<customerid>`.

For example, to register `abcxyz.com`, you need to create a DNS TXT record for `_sap_mobsec.abcxyz.com` with the value `customerid=<customer_ID>`, where `<customer_ID>` is the value shown in the Register Email Domain page of the `abcxyz` account's Configuration Discovery settings page.

4. Click *Save*.

If the domain is valid, it appears in a list at the application level.

5. (Optional) To delete a domain, select *Delete*.

Related Information

[Adding Onboarding Codes \[page 181\]](#)

[Adding Application Configurations \[page 182\]](#)

[Enabling Applications to Discover Configurations \[page 179\]](#)

1.5.5.2.2 Adding Onboarding Codes




Onboarding codes can be added to the application as an alternative to Service provider domains.

Context

Note

- Reduces the dependency on other individuals or processes making it easier to use onboarding codes to setup and use Discovery Service.
- It is an alternative to email domain, useful in solutions where the customer's email domain is not known.
- It requires you to explicitly inform the users about the onboarding code. For example, you can email it to users or post it on the portal etc.

Procedure

1. In Mobile Services cockpit, select  [Settings](#)  [Discovery Service](#) .
2. Select the plus sign.
3. Specify the description

Note

Description enables you to identify the code that is being created.

4. Click [Save](#).

Codes appear in a list at the application level.

Related Information

[Enabling Applications to Discover Configurations \[page 179\]](#)

[Adding Service Provider Domains \[page 180\]](#)

1.5.5.2.3 Adding Application Configurations

Application configurations can be added at the application level.

Context





You can either add the domain or the onboarding code along with the configuration data to each app. You can add the same application configuration to multiple domains.

Note that once the application configuration is published, if you update the multi-user property, it will not affect the published configuration value. If you want the published application configuration to use the latest multi-user property value, you must delete the saved domain and onboarding code configuration from the application [Discovery](#) tab, and then recreate them.

Note

Do not delete the domain and onboarding code from the ► [Settings](#) ► [Discovery Service](#) ► page.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select an application, then select [Discovery](#).
3. To add a domain based configuration to an application:
 1. Select the plus sign under [Domains](#).
 2. Select the domain from the list.
 3. Select [Default](#) to use the default application configuration and [Save](#).
 4. Select  and select [Custom](#) to customize the generated default configuration and [Save](#).
 5. Select  to delete a default or custom configuration.
4. To add onboarding codes based configuration to an application:
 1. Select the plus sign under [Onboarding Codes](#).
 2. Select the code from the list.
 3. Select [Default](#) to use the default application configuration and [Save](#).
 4. Select  and select [Custom](#) to customize the generated default configuration and [Save](#).
 5. Select  to delete a default or custom configuration.

Related Information

[Enabling Applications to Discover Configurations \[page 179\]](#)

[Adding Service Provider Domains \[page 180\]](#)

[Adding Onboarding Codes \[page 181\]](#)

1.5.5.3 Managing SAP Build Apps

Configure and manage an SAP Build App in Mobile Services cockpit.

Prerequisites

Currently, you cannot create SAP Build Apps from Mobile Services cockpit. The `buildApps` application type can only be created by the build apps team from their service.

Context

SAP Build Apps provides an option for deploying a scaled down mobile app and managing it using the cockpit. Currently the `buildApps` application type can only be created by the build apps team from their service. Once it is created, the application appears in the Mobile Services cockpit *SAP Build Apps* application list. Note that limited features and functions are available for `buildApps` applications. Only features that are available to SAP Build appear in the cockpit.

Procedure

1. In Mobile Services cockpit, select  *Mobile Applications*  *SAP Build Apps* .

You can view a list of applications with summary information such as application ID, application name, application type, vendor, application state, whether outdated, and the creation date. Use the search and sort options to locate the applications you're interested in.

SAP Build Apps Application Summary

Column	Description
Application ID	The unique application identifier.
Name	The application name.
Application Type	SAP Build Apps

Column	Description
Vendor	The vendor who supplied the application.
State	<p>The current state of the application such as:</p> <ul style="list-style-type: none"> Started – the application is running normally. Inconsistent – applications may become inconsistent if they are deleted or changed from the SAP BTP cockpit. From Mobile Services cockpit, you can either delete or restore the application. Deleted from SAP BTP cockpit – the application was deleted. You can restore or delete it in the Mobile Services cockpit. Restoring – the application is being restored from an inconsistent state.
Outdated	Whether the application is outdated. If checked, you are prompted to update the application to a newer version that contains fixes and improvements. Typically you should update the application.
Creation Date (UTC+0000)	The application creation date, either in UTC or local format, depending on what you selected for your User Setting.

2. Select an application to view or modify its details, as described in *Configuring SAP Build Apps*.
3. Select *Import* to import a new app.
 1. In *Import Application*, enter a file name or select *Browse* to locate it.
 2. Click *Save*.
The app is imported and added to the list of applications..
 3. Configure the app as described in *Configuring SAP Build Apps*.

[Configuring SAP Build Apps \[page 184\]](#)

Configure and edit an SAP Build Apps application and manage it from Mobile Services cockpit.

1.5.5.3.1 Configuring SAP Build Apps

Configure and edit an SAP Build Apps application and manage it from Mobile Services cockpit.

Context


Only features that are available to SAP Build appear in the cockpit.


Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [SAP Build Apps](#).
2. Select an application to view its details.
3. Configure or modify the SAP Build Apps application. To prevent accidental changes, you can lock an application once its configurations are complete. To make changes, you'll need to manually unlock the application.
 - [Info](#) provides a summary of settings for the selected application.
Under [Application Details](#) is a summary of client app information. You can edit some of the values. The list under [Assigned Features](#) identifies the features that are currently assigned to the application. You cannot add or delete features. For more information, see [Configuring Assigned Features](#). In the Cloud Foundry environment you can also configure [Custom Routes](#).
 - [APIs](#) provides onboarding and API features.
 - Under [Onboarding](#), you can find one or more QR codes that can be used to configure the mobile application to connect to this Mobile Services instance.
For [In-app Scanning Code](#), the device user must first manually launch the app, then scan the QR code from within the app to initiate the onboarding process.
For [Default Configuration](#), native app developers can download a native-specific file containing the Discovery Service default configuration. Select [iOS](#) to download the configuration file in `plist` format, or select [Android](#) to download the file in `JSON` format. Simply add the file to your iOS or Android app project without further formatting.
For [Configuration](#), you can view any application routes that you've associated with the application. Select edit to make changes. See [Defining Applications \(Cloud Foundry\)](#) for information about the XSUAA property, [Domain of Application Route](#).
For [Apple and Android Launch App Code](#), the device user can select the URL (or scan the QR code for Apple's Camera App or an Android application that allows reading QR codes) to find and launch the application installed locally. If the user enables Android app links, it also supports Android's camera app. The App-launch QR code is generated if you enable Apple Universal Links or Android App Links or both of them in the [Application Links](#) section; the label text of the QR code depends on which application links are enabled ("Apple launch app code," "Android launch app code," or "Apple and Android launch app code."
For [Apple and Android Launch App Code with URI Scheme](#), the device user can scan the QR code or select the URL, to use an appLink URL to find and launch the application. This option only appears if you enabled Device Application URI Schemes in the [Application Links](#) tab. You must select one Device Type, and input the corresponding device application URI Scheme for iOS or Android.

Note

(Mobile Development Kit only) If you set up Apple and Android Device Application URI schemes in the [Application Links](#) tab, you see two separate QR codes, labeled [Apple Launch App Code with URI Scheme](#) and [Android Launch App Code with URI Scheme](#)

If the app security configuration changes, you must generate a new QR code. Also, please note that the URL can be very long and might be longer than the value shown on the screen. Use [Copy to Clipboard](#) () , to copy its complete value.

Under [API](#), you can view a list of frequently used API URLs. Use [Copy to Clipboard](#) () , to copy read-only URLs to the clipboard, to paste elsewhere, avoiding typing errors and ensuring accuracy.

Note

The Server URL is also the `<mobile_services_host>` name.

- You can use [Application Links](#) to enable universal links for Apple and application links for Android apps. Apple universal links and Android application links enable users to open an application locally without downloading the app or using a browser container.
- [Users](#) enables you to export or delete data for a user, and to block one or more users from using a SAP Build Apps application
- [Security](#) enables you to manage security for the selected application. For details see *Configuring App Security* and *Configuring IAS Security*.
- [Log Settings](#) enables event logs at the application level. You can specify any of the individual services that are assigned to the app, as well as Mobile AppRouter. This gives you more control over the event log information you want to see at the application level. See *Enabling Event Logs for Apps*. You can continue to view event logs as described in *Viewing Event Logs*.

1.5.5.4 Configuring Micro Apps

Note

Micro App support is available only in select countries/regions.

You can use Mobile Services cockpit templates to configure micro apps, including WeCom (enterprise), WeChat (social) and DingTalk, to take advantage of SAP Mobile Services features.

[Managing Micro Apps \[page 187\]](#)

Use the mobile services cockpit to manage Micro Apps, such as WeCom (enterprise), WeChat (social) and DingTalk.

[Defining Micro Apps \[page 188\]](#)

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

[Configuring Assigned Features for Micro Apps \[page 192\]](#)

Configure features associated with a Micro App definition.

[Configuring WeCom Settings \[page 193\]](#)

Configure settings for a WeCom micro app.

[Configuring WeChat Settings \[page 194\]](#)

Configure settings for a WeChat micro app.

[Configuring DingTalk Settings \[page 196\]](#)

Configure settings for a DingTalk micro app.

[Configure Destinations for DingTalk Service \[page 197\]](#)

The DingTalk H5 Application requires you to set different exit node IPs for different customers.

[Micro App APIs \[page 199\]](#)

A list of configured APIs for micro apps.

[Defining Micro App Security \[page 204\]](#)

Define the settings that control user authentication behavior for the selected micro app.

[Importing Micro App Configurations \[page 205\]](#)

Import a Micro App configuration from one SAP Mobile Services environment to another. Many configuration settings are retained, but you must reconfigure some application settings for the target server environment.

[Managing Micro App User Registrations \[page 205\]](#)

Manage multiple user registrations for a micro app. Registrations are associated with an authenticated or an anonymous user on one or more devices.

[Sending Micro App Notifications \[page 206\]](#)

(WeCom and DingTalk) Send different types of notifications to a registered Micro App user.

1.5.5.4.1 Managing Micro Apps

Use the mobile services cockpit to manage Micro Apps, such as WeCom (enterprise), WeChat (social) and DingTalk.

Context

Micro App support is available only in select countries/regions.

Procedure

1. In the Mobile Services cockpit, select  [Mobile Applications](#)  [Micro App](#) .

You can view a list of apps with summary information such as application ID, name, application type, vendor and so forth. Use the search and sort options to locate the apps in which you're interested.

Micro App Summary

Column	Description
Application ID	A unique identifier for the Micro App.
Name	The Micro App name.
Application Type	The Micro App type, such as WeCom, WeChat or DingTalk.
Vendor	The vendor that supplied the Micro App.
License Type	The application plan that was in place when the Micro App was created.
State	The current state of the Micro App, such as Started, Inconsistent, Deleted or Restoring.

Column	Description
Outdated	Whether the Micro App is outdated. If checked, you are prompted to update the Micro App to a newer version that contains fixes and improvements. Typically you should update the Micro App.
Creation Date (UTC/local)	The Micro App creation date, either in UTC or local format, depending on what you selected for your User Setting.

2. Select a Micro App to view its details, and make changes.

- [Info](#) provides a summary of settings for the selected Micro App. The list under [Assigned Features](#) identifies the features that are currently assigned to the Micro App, and enables you to add and delete more. You can also create custom routes.
- [APIs](#) provides a summary of some configuration options (select edit to make changes), and a list of frequently used API URLs (use [Copy to Clipboard](#) to copy read-only URLs to the clipboard). Note that the Server URL is also the `<mobile_services_host>` name.
- [Users](#) (WeChat only) enables you to export or delete data for a user, and to block one or more users from using a Micro App.
- [Security](#) enables you to manage security for the selected Micro App as described in *Defining Micro App Security*. See *Configuring App Security* for additional security details.
- [Alert](#) lets you enable alerts for WeChat apps and establish the message threshold and schedule for alerts. To receive notifications, configure Subscriptions and Alerts through the Alert Notification service (subscriptions and alerts cannot be managed through the cockpit for the Mobile Services). See *Configuring Alert Settings* and *Subscribe to Mobile Service Alerts* for information.
- [Log Settings](#) is used to enable event logs for the Micro App. You can specify any of the individual services that are assigned to the Micro App, as well as Mobile Application. This gives you more control over the event log information you want to include. See *Enabling Event Logs for Apps* and *Viewing Event Logs* for information.

1.5.5.4.2 Defining Micro Apps

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

Context

After you define the application, configure the assigned features, or add more features. See *Configuring Assigned Features for Micro Apps*.

Procedure

1. In the Mobile Services cockpit, select ► *Mobile Applications* ► *Micro App* ►, then select *New*.
2. In *New Application*, enter:

Field	Value
Config Templates	Select a configuration template type: <ul style="list-style-type: none">• DingTalk• WeChat (social)• WeCom (enterprise)

Field	Value
ID	<p>Unique identifier for the application, in reverse-domain notation. This is the application or bundled identifier that is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Mobile Services, and client applications use the application ID when sending requests to the server.</p> <p>An application ID:</p> <ul style="list-style-type: none"> • Must be unique • Must start with an alphabetic character • Can contain only alphanumeric characters, underscores, and periods • Can contain up to 64 characters • Cannot include spaces • Cannot begin with a period, and cannot contain two consecutive periods • Cannot be any of these case-sensitive keywords: <code>Admin</code>, <code>AdminData</code>, <code>Pushsmp_cloudresource</code>, <code>test-resources</code>, <code>resources</code>, <code>Scheduler</code>, <code>odata</code>, <code>applications</code>, <code>Connections</code>, <code>public</code>, <code>lcm</code> <p>We recommend that you assign IDs that contain a minimum of two periods, for example, <code>com.sap.mobile.app1</code>.</p> <p>If you are building an Android application, its ID must follow the Google defined rules, or the Android build fails:</p> <ul style="list-style-type: none"> • The ID must have at least two segments (one or more periods). • Each segment must start with a letter. • All characters must be alphanumeric or an underscore <code>[a-zA-Z0-9_]</code>. <p>The prefix "com.sap.webide" is reserved for packaged apps created from WebIDE.</p>
Name	The application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.
CorpID	(WeCom, DingTalk - required) Unique ID of the WeCom or DingTalk enterprise.
AgentID	(WeCom, DingTalk - required) Unique ID of every enterprise application.

Field	Value
WeChat AppID	(WeChat only - required) Unique ID for the WeChat application.
Description	(Optional) The description can contain up to 255 alphanumeric and special characters.
Vendor	(Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long.
License Type	<p>The license type that you want to use for your account: "standard" (default), "b2c", "free" and "resources"; "lite" is available only for trial subaccounts. The license determines capabilities and charges.</p> <p>You can change your service plan from "free" to "standard", "b2c" or "resources", enabling you to migrate your app without losing your work.</p> <div> <p>Note</p> <p>Changing service plans is not reversible. You cannot switch from "standard", "b2c" or "resources" to "free".</p> </div> <p>See <i>Service Plans</i> for additional information.</p>
Domain of Application Route	<p>In regions that support multiple domains, select the application route through which end users can reach the application. Routes are associated with a space, and are configured in SAP BTP cockpit. The drop-down list shows all shared and custom domains that are available for the customer organization.</p> <div> <p>Note</p> <p>If you want to use a custom domain, you can create one according to information in What Is Custom Domain.</p> </div>

3. Select [Next](#) to continue.

Field	Value
XSUAA Service	<p>Select the XSUAA authentication and authorization service to use. You can select Default Instance to create a new instance, or you can select an existing service from the list.</p> <p>Using an existing XSUAA instance is useful if you already have a Cloud Foundry application deployed and want to connect from Mobile Services to this application. In this case it can be handy to re-use an existing XSUAA instance.</p> <p>Select the default instance for other scenarios.</p>
xs-security.json	A JSON file that defines the authentication methods and authorization types used to access your application. See Application Security Descriptor Configuration Syntax , for additional information.
Token Lifetime	Enter the token lifetime, and select the units (Days, Hours, or Minutes).
Refresh Token Lifetime	Enter the refresh token lifetime, and select the units (Days, Hours, or Minutes).
Approved Providers	Indicate whether the app should support all approved providers or only selected providers.
System Attributes in Token	Indicate groups, role collections or both.

- Click [Finish](#) and [OK](#) to confirm. The [Info](#) page appears with current settings.

1.5.5.4.3 Configuring Assigned Features for Micro Apps


Configure features associated with a Micro App definition.

Context

You can add additional features, and delete features that are not required. Some features require coding in the application. In general, there should be no dependency between features.

Procedure




- In the Mobile Services cockpit, select [Mobile Applications](#) [Micro App](#).
- Select an application and access the [Info](#) tab.
- Under [Assigned Features](#), you can view the list of features assigned to the selected application. These features are included by default:









- Mobile Connectivity – define a data endpoint for the application, enable the detailed event log, and view feature details.
 - Mobile Micro App – define DingTalk or WeCom or WeChat settings and service keys, and view feature details.
 - Mobile Settings Exchange – manage user registrations, send notifications, view feature details, and export mobile services data in JSON format. For WeChat only, you can see WeChat User Mapping information for the selected user.
4. (Optional) Under *Assigned Features*, select the add icon  to add one of the following features:
- Mobile Sample OData ESPM – use the sample service during development and testing.
 - Mobile Network Trace – view network trace information collected from mobile applications for debugging.
- Then select *OK*.
5. Select an assigned feature, such as Mobile Micro App, to configure its settings, and then select *Save*.
6. (Optional) Select *Reset* to reset settings to default values.
7. (Optional) Select *Remove from Application* to remove the feature from the application.

1.5.5.4.4 Configuring WeCom Settings

Configure settings for a WeCom micro app.

Procedure

1. In the Mobile Services cockpit, select  *Mobile Applications*  *Micro App* , then choose an existing WeCom app or create one.
2. Select *Mobile Micro App* under *Assigned Features*.
3. From the *Configuration* tab, define the following settings:

Name	Description
CorpID	(Required) Unique ID of the WeCom enterprise account. Find this information under  <i>WeCom Admin Panel</i>  <i>My Enterprise</i>  <i>Enterprise Info</i>  .
AgentID	(Required) Unique ID of an enterprise application. Find this information under  <i>WeCom Admin Panel</i>  <i>Enterprise Application</i>  <i>Into the Application</i>  .

Name	Description
App Secret	(Required) Enterprise application secret. Find this information under ► WeCom Admin Panel ► Enterprise Application ► Into the Application ►.
Message Handler Endpoint	Select the message handler endpoint from connection list. Copy the value (Message Handler URL) to edit the API receiving setting URL in the WeCom Admin Panel .
Message Handler URL	The Message Handler URL appears, based on the Message Handler Endpoint you selected. Use the copy icon to copy the URL to the clipboard, to paste elsewhere.
Token	(Required) WeCom enterprise application message receiving token. Find this information in the ► WeCom Admin Panel ► Specific App ► Message Receiving ► API Receiving Setting ►.
EncodingAESKey	(Required) WeCom enterprise application message receiving message body encoding key. Find this information in the ► WeCom Admin Panel ► Specific App ► Message Receiving ► API Receiving Setting ►.

4. From the [Service Keys](#) tab, for some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials. The feature must be able to support service keys. See [Service Keys](#).
5. On the [Info](#) tab, you can find additional information, including the feature description and documentation and support URLs.

Related Information

[Service Keys \[page 303\]](#)

1.5.5.4.5 Configuring WeChat Settings

Configure settings for a WeChat micro app.

Context

Procedure

1. In the Mobile Services cockpit, select ► [Mobile Applications](#) ► [Micro App](#) , then choose an existing WeChat app or create one.
2. Select [Mobile Micro App](#) under [Assigned Features](#).
3. From the [Configuration](#) tab, define the following settings:

Name	Description
WeChat AppID	(Required) Unique ID for the WeChat application. In the WeChat Official Account Platform, after applying for an Official Account, you can find the WeChat AppID in Basic Configuration.
App Secret	(Required) The application secret for the WeChat application. In the WeChat Official Account Platform, after applying for an Official Account, you can find the App Secret in Basic Configuration.
Message Handler Endpoint	Select the message handler endpoint description from the connection list.
Message Handler URL	The Message Handler URL appears, based on the Message Handler Endpoint you selected. Use the copy icon to copy the URL to the clipboard, to paste elsewhere.
Token	(Required) The WeChat application message receiving token. Token and EncodingAESKey are used for developer server verification of message and autonomous settings in the WeChat Official Account Platform. In the WeChat Official Account Platform, you can find the Token information in Basic Configuration.
EncodingAESKey	The WeChat application message receiving message body encoding key. Token and EncodingAESKey are used for developer server verification of message and autonomous settings in the WeChat Official Account Platform. In the WeChat Official Account Platform, you can find the EncodingAESKey information in Basic Configuration.

4. From the [Service Keys](#) tab, for some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials. See [Service Keys](#) for additional information.
5. On the [Info](#) tab, you can find additional information, including the feature description and documentation and support URLs.

1.5.5.4.6 Configuring DingTalk Settings

Configure settings for a DingTalk micro app.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Micro App](#). You can view existing DingTalk apps or create one.

To create a new DingTalk apps, you must select the *DingTalk* configuration template.

2. Select [Mobile Micro App](#) under *Assigned Features*.
3. From the *Configuration* tab, define the following settings:

Name	Description
CorpID	(Required) Unique ID of the DingTalk enterprise account. Find "CorpID" in DingTalk Development Platform Account Management .
AgentID	(Required) Unique ID of an enterprise application. Find this information in DingTalk Web Admin Panel > Self-built application > Application Info .
App Key	(Required) Enterprise application key. This value is auto-generated when the developer creates the application. Find this information in DingTalk Web Admin Panel > Self-built application > Application Info .
App Secret	(Required) Enterprise application secret. This value is auto-generated when the developer creates the application. Find this information in DingTalk Web Admin Panel > Self-built application > Application Info .

4. From the *Service Keys* tab, for some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials. The feature must be able to support service keys. See *Service Keys*.
5. On the *Info* tab, you can find additional information, including the feature description and documentation and support URLs.

Related Information

[Service Keys \[page 303\]](#)

1.5.5.4.7 Configure Destinations for DingTalk Service

The DingTalk H5 Application requires you to set different exit node IPs for different customers.

Destination service is used to distinguish between different customers in the same landscape. Users can either use the SAP Cloud Connector as a proxy server or create their own HTTP proxy server.

[Configuring Destinations using the SAP Cloud Connector \[page 197\]](#)

Users can configure destinations by using the SAP Cloud Connector as a proxy server.



[Configuring Destinations using HTTP Proxy \[page 198\]](#)






Users can configure destinations by creating their own proxy server.

1.5.5.4.7.1 Configuring Destinations using the SAP Cloud Connector

Users can configure destinations by using the SAP Cloud Connector as a proxy server.

Procedure

1. In the SAP Cloud Connector cockpit, select your *Subaccount*.
 2. Select *Cloud To On-Premise* on the left navigation panel.
 3. Select *Access Control* and click the Add icon .
 4. In the Add System Mapping dialog:
 - Select **Non-SAP System** as the *Back-end Type* and click *Next*.
 - Select **HTTPS** as the *Protocol* and click *Next*.
 - Enter **oapi.dingtalk.com** as the *Internal Host*, **443** as the *Internal Port*, and click *Next*.
 - Enter **oapi.dingtalk.local** as the *Virtual Host*, **80** as the *Virtual Port*, and click *Next*.
-  **Note**

The virtual host name can be anything other than "oapi.dingtalk.com".
- Select **Use Internal Host** as the *Host in Request Header* and click *Next*.
 - Verify the mapping and click *Finish*.
5. Select the mapping you created and click the Add icon  to add the resources.
6. In the Add Resource dialog, enter **/** as the *URL Path*, select *Path And All Sub Paths* as the *Access Policy*, and click *Save*.
7. In any web browser, open **https://oapi.dingtalk.com**, click the lock icon and save the cert file.
8. Select *Configuration* on the left navigation panel.
9. Select  **ON PREMISE**  **Trust Store**  and click the Add icon .

10. In the Add Public Key dialog:

- Click [Browse](#).
- Select the cert you downloaded and click [Save](#).

11. In the SAP BTP cockpit, select your [Subaccount](#).

12. Select [Destinations](#) on the left navigation panel and click [New Destination](#).

13. Under Destination Configuration, enter:

Properties

Field	Value
Name	dingtalk
Type	HTTP
URL	http://oapi.dingtalk.local
<div><div>📘 Note</div><div>The URL is http://<virtual host name></div></div>	
Proxy Type	OnPremise
Authentication	NoAuthentication
Location ID	Your SAP Cloud Connector location ID
MobileEnabled	True

14. Click [Save](#).

1.5.5.4.7.2 Configuring Destinations using HTTP Proxy

Users can configure destinations by creating their own proxy server.

Procedure

1. In the SAP BTP cockpit, select your [Subaccount](#).
2. Select [Destinations](#) on the left navigation panel and click [New Destination](#).
3. In the Destination Configuration dialog, enter:

Properties

Field	Value
Name	dingtalk
Type	HTTP
URL	Specify your http proxy server URL

Field	Value
Proxy Type	Internet
Authentication	NoAuthentication
MobileEnabled	True
Use default JDK truststore	Select the checkbox to enable.

4. Click [Save](#).

1.5.5.4.8 Micro App APIs

A list of configured APIs for micro apps.

Procedure

1. In the Mobile Services cockpit, select [Mobile Applications](#) [Micro App](#), select an application, and then choose the [APIs](#) tab.
2. Under [API](#), you can view a list of frequently used API URLs.
 - Of special interest for WeCom:
 - WeCom Message Handler – the API used to control messaging. This API appears only after the message handler endpoint has been configured in [WeCom Settings](#).
 - WeCom Push – the API used to control push notifications.
 See *Micro App APIs for WeCom* for details.
 - Of special interest for WeChat:
 - WeChat Message Handler – the API used to control messaging. This API appears only after the message handler endpoint has been configured in [WeChat Settings](#).
 - Of special interest for DingTalk:
 - DingTalk Push – the API used to control push notifications.

Related Information

[Micro App APIs for WeCom \[page 200\]](#)

1.5.5.4.8.1 Micro App APIs for WeCom

Use the WeCom Message Handler and WeCom Push APIs for stand-alone WeCom Java application development and deployment to the Cloud Foundry environment.

- **WeCom Message Handler** – the API used to control WeCom message receiving. This API appears only after the message handler endpoint has been configured in WeCom settings.
- **WeCom Push** – the API used to control push notifications.

WeCom Message Handler API

This API is used to receive XML WeCom message from WeCom end users through WeCom Server. The XML payload received from WeCom Server is documented in [WeCom Doc](#) 🇨🇳 (Chinese Only).

The response payload to the WeCom Server (which will be pushed back to the WeCom end users) should also follow the same XML format as receiving payload.

WeCom Push API

We are supporting different message types for WeCom message Push feature. You can make the POST call to the WeCom Push API with following information for pushing WeCom messages to end user(s).

- header `X-API-Key`: this is the API key generated from [Service Keys](#) tab of [WeCom Settings](#).
- header `Content-Type`: `application/json`
- request body: the JSON message payload.

The JSON payload examples are provided in the following sections.

text

```
{
  "notification": {
    "wechat": {
      "createtime": "2019-12-17T06:24:26.451Z",
      "text": {
        "content": "hello world!"
      },
      "msgtype": "text"
    }
  },
  "users": ["P000001"]
}
```

textcard

```
{
  "notification": {
    "wechat": {
      "createtime": "2019-12-17T06:24:58.808Z",
      "textcard": {
```



```

        "btntext": "MoreInfo",
        "description": "Test for textcard msg type",
        "title": "title_textcard",
        "url": "http://www.abc.com"
    },
    "msgtype": "textcard"
},
{
    "users": ["P000001"]
}

```

markdown

```

{
  "notification": {
    "wechat": {
      "createtime": "2019-12-17T06:25:02.936Z",
      "markdown": {
        "content": "Meeting room already reserved.
rn>**items*>Meeting<font color=" info ">meeting1</font> rn >john@abcdef rn>If
required, please click [modify resource](https://abcde.com)"
      },
      "msgtype": "markdown"
    }
  },
  "users": ["P000001"]
}

```

image

The image file should be base64 encoded.

- **Image file size limit:** 2 MB
- **Supported format:** ".jpg", ".png"

```

{
  "notification": {
    "wechat": {
      "image": {
        "media_file_encoded": "data:image/jpeg;base64,XXXXXXXXXXXXXXXX",
        "media_file_name": "testimage.jpg"
      },
      "createtime": "2019-12-17T06:25:09.509Z",
      "msgtype": "image"
    }
  },
  "users": ["P000001"]
}

```

voice

The voice file should be base64 encoded.

- **Voice file size limit:** 2 MB
- **Voice time limit:** 60 seconds
- **Supporting format:** only supporting ".amr"

```

{
  "notification": {
    "wechat": {
      "voice": {
        "media_file_encoded": "data:application/octet-
stream;base64,XXXXXXXXXXXXXXXX",

```

```

        "media_file_name": "test.amr"
      },
      "createtime": "2019-12-17T06:52:14.515Z",
      "msgtype": "voice"
    },
    "users": ["P000001"]
  }
}

```

video

Video file should be base64 encoded.

- **Video file size limit:** 10 MB
- **Supporting format:** only supporting ".mp4"

```

{
  "notification": {
    "wechat": {
      "createtime": "2019-12-17T06:52:19.631Z",
      "video": {
        "media_file_encoded": "data:video/mp4;base64,XXXXXXXXXXXXXXXXXXXX",
        "media_file_name": "test.mp4",
        "description": "video description",
        "title": "Video Title"
      },
      "msgtype": "video"
    },
    "users": ["P000001"]
  }
}

```

file

Push general file to user(s), the file should be base64 encoded.

- File size limit: 20 MB
- No file extension name limit

```

{
  "notification": {
    "wechat": {
      "createtime": "2019-12-17T06:52:23.417Z",
      "file": {
        "media_file_encoded": "data:application/x-zip-compressed;base64,XXXXXXXXXX",
        "media_file_name": "abc.zip"
      },
      "msgtype": "file"
    },
    "users": ["P000001"]
  }
}

```

WeCom news

```

{
  "notification": {
    "wechat": {
      "news": {
        "articles": [{
          "picurl": "https://abc/def/pic1.jpg",
          "description": "description for news1",

```

```

        "title": "News1",
        "url": "http://www.wechatnews1.com"
    }, {
        "picurl": "https://abc/def/pic2.jpg",
        "description": "description for news2",
        "title": "News2",
        "url": "http://www.wechatnews2.com"
    }, {
        "picurl": "https://abc/def/pic3.jpg",
        "description": "description for news3",
        "title": "News3",
        "url": "http://www.wechatnews3.com"
    }
    ],
    "createtime": "2019-12-17T06:52:25.765Z",
    "msgtype": "news"
},
{
    "users": ["P000001"]
}

```

WeCom mpnews

Thumb image file should be base64 encoded.

- The limit of thumb image file is same as the limit of Image message.

```

{
    "notification": {
        "wechat": {
            "createtime": "2019-12-17T06:52:28.337Z",
            "mpnews": {
                "articles": [{
                    "thumb_media_file_name": "pic1.jpg",
                    "thumb_media_file_encoded": "data:image/
jpeg;base64,XXXXXXXXXX",
                    "author": "user1",
                    "digest": "digest1",
                    "content_source_url": "http://www.wechatmpnews1.com",
                    "title": "Title1",
                    "content": "content1"
                }, {
                    "thumb_media_file_name": "pic2.jpg",
                    "thumb_media_file_encoded": "data:image/
jpeg;base64,XXXXXXXXXX",
                    "author": "user2",
                    "digest": "digest2",
                    "content_source_url": "http://www.wechatmpnews2.com",
                    "title": "Title2",
                    "content": "content2"
                }
            ]
        },
        "msgtype": "mpnews"
    },
    "users": ["P000001"]
}

```

1.5.5.4.9 Defining Micro App Security




Define the settings that control user authentication behavior for the selected micro app.

Prerequisites

Default and custom trust configuration must already be established in SAP BTP cockpit.

For information about establishing trust, see *Configuring Security Trust* and [Establish Trust and Federation with UAA Using Any SAML Identity Provider](#).

Procedure

1. In the Mobile Services cockpit, select  [Mobile Applications](#)  [Micro App](#) .
2. Select an application, then select [Security](#) to configure application security.
3. Under [Application Settings](#), select . Configure application-level security settings:

Application Settings

Field	Value
CSRF Protection	Enables Cross-Site Request Forgery (CSRF) protection. Requires a CSRF token to work.
Security Configuration	Select the appropriate security configuration: <ul style="list-style-type: none">• WeCom OAuth – WeCom-specific token-based authentication. Selected automatically for WeCom apps.• WeChat OAuth – WeChat-specific token-based authentication. Selected automatically for WeChat apps.• DingTalk OAuth – DingTalk-specific token-based authentication. Selected automatically for DingTalk apps.

4. Under [WeCom Settings](#), view the current security settings.
5. Under [WeChat Settings](#), view the current security settings.
6. Under [DingTalk Settings](#), view the current security settings.
7. For other security settings, such as [XSUAA Settings](#), [Role Settings](#), [Anonymous Access](#), [Allowed IP/CIDR](#), [Cross Domain Access](#), see [Configuring App Security](#) for details.

Related Information

[Configuring Security Trust \[page 294\]](#)

1.5.5.4.10 Importing Micro App Configurations

Import a Micro App configuration from one SAP Mobile Services environment to another. Many configuration settings are retained, but you must reconfigure some application settings for the target server environment.

Procedure

1. In the Mobile Services cockpit, select ► [Mobile Applications](#) ► [Micro App](#) ►, then select [Import](#) in the upper right corner.
2. In [Import Application](#), click in the blank field or select [Browse](#); choose a file name in the pop-up dialog; and then select [Save](#).

If you import a Micro App with the same ID as an existing one, you see the error message `Application already exists`.

Next Steps

Update the Micro App by configuring required WeCom, WeChat, or DingTalk settings.

1.5.5.4.11 Managing Micro App User Registrations

Manage multiple user registrations for a micro app. Registrations are associated with an authenticated or an anonymous user on one or more devices.

Procedure

1. In the Mobile Services cockpit, select ► [Mobile Applications](#) ► [Micro App](#) ►, then select the application.
2. Select [Mobile Settings Exchange](#) under [Assigned Features](#), then [User Registrations](#).
3. Under [Automatic Removal](#), you can enable automatic user removal, and indicate the time frame for removal in days.
4. Under [Summary](#), view current number of user registrations.
5. Under [Registered Users](#), you can filter and sort user registrations.

Column Descriptions

Column	Description
Registration ID	Unique identifier provided by the client application, or system-generated application registration ID.

Column	Description
User Name	User name identified with the registered application.
Last Connection	The date and time when the application was registered, in the format YYYY-MM-DD HH:MM:SS. The value appears in local or UTC format, depending on your user setting.
Actions	You can delete the user registration.

Note

For some push providers, when you delete user registrations for auto registration in [Mobile Settings Exchange](#), you must also manually delete the device registration from ► [Mobile Push Notifications](#) ► [Push Notifications](#) ►. This is because the provider does not provide the information needed for automatic deletion in push. For these providers, you must perform the manual delete: Baidu, Custom, and W3C.

6. Select a registered user to view details. Categories may vary by application type.
 - [User Registration](#) – the unique ID assigned to the registered user.
 - [Info](#) – basic information about the registered user.
 - [Device](#) – known information about the device used by the registered user.
 - (WeChat only) [WeChat User Mapping](#) – the WeChat identifier, nickname, and profile photo, if provided by the registered user.
 - [Push](#) – any push notification settings for the registered user.
 - [Custom](#) – any custom items configured for the registered user.
7. (WeCom and DingTalk) Select a registered user, then [Send Notifications](#) to send notification alerts to the user. See [Sending Micro App Notifications](#).

Related Information

[Sending Micro App Notifications \[page 206\]](#)

1.5.5.4.12 Sending Micro App Notifications

(WeCom and DingTalk) Send different types of notifications to a registered Micro App user.

Context

For WeCom and DingTalk micro apps, the supported message types are:

- Text

- File
- Image
- Markdown
- News
- Text Card
- Video
- Voice
- WeCom News

For the [Markdown](#) option, a subset of the markdown syntax is supported:

- **Headings** – heading levels 1 to 6:

```
# Heading level 1
## Heading level 2
### Heading level 3
#### Heading level 4
##### Heading level 5
##### Heading level 6
```

- **Bold** – in the format:

```
**bold**
```

- **Links** – enclose link name in brackets, for example:

```
[This is a link name](http://work.weixin.qq.com/api/doc)
```

- **Code** – enclose in single quotes (one line only), in the format:

```
`code`
```

- **Blockquotes** – in the format:

```
> Quotes
```

- **Fontcolor** – up to three colors are supported:

```
<font color="info">Green</font>
<font color="comment">Gray</font>
<font color="warning">Orange</font>
```

For details about the supported markdown elements, see:

- [WeCom Supported Markdown Syntax](#) 🌐 (Chinese only)
- [Markdown Guide](https://www.markdownguide.org/basic-syntax) - Access using <https://www.markdownguide.org/basic-syntax>

Procedure

1. In the Mobile Services cockpit, select **Mobile Applications** > **Micro App**, then select the application.
2. Select **Mobile Settings Exchange** under **Assigned Features**, then **User Registration**.
3. Under **Registered Users**, use the filter and sort options to select one or more user registrations as described in *Managing Micro App User Registrations*.

4. Under [Registered Users](#), select a registered user, then select [Send Notifications](#) to send notification alerts to the user.

In [Send WeCom / DingTalk Push Notification](#), select a message type.

For [File](#), select a file to send that is no larger than 20MB.

1. Select [Browse](#) and navigate to the file.
2. Select the file, and then [Open](#).

For [Image](#), select a .PNG or .JPG file that is no larger than 2MB.

1. Select [Browse](#) and navigate to the image file.
2. Select the file, and then [Open](#).

For [Markdown](#), select a markdown file to send. The file must follow the guidelines described above.

1. Select [Browse](#) and navigate to the .md file.
2. Select the file, and then [Open](#).

For [News](#), specify one or more articles to send:

1. Select [+](#) to add one or more news articles.
2. Provide information to identify the article.

News Article Properties

Properties	Description
Title	The article title.
Description	A description for the article.
URL	The article URL.
News Cover URL	The news cover URL.
Action	You can delete the article.

For text messages, select [Text](#), and compose the message text to send.

For text cards, select [Text Card](#), and define the text card properties:

Text Card Properties

Properties	Description
Title	The text card title.
Description	A description for the text card.
URL	The data endpoint URL for the text card.
Button Text	The text to appear on the text card button.

For [Video](#), select an .MP4 file that is no larger than 10MB, and define the video properties.

Video Properties

Properties	Description
Title	The video title.

Properties	Description
Description	A description for the video.

For [Voice](#), select an .AMR file that is no larger than 2MB and no longer than 60 seconds.

1. Select [Browse](#) and navigate to the video file.
2. Select the file, and then [Open](#).

For [WeCom News](#), send up to eight WeCom news articles in a batch. Each article can include a cover picture, a source URL, and other information. If you change your mind, you can easily delete one or more of the articles before you send it.

Properties	Description
Title	The title for the selected articles.
Cover Picture	Browse to select a .PNG or .JPG image file no larger than 2MB.
Author	The author name.
Content source URL	Provide the URL for the content source.
Content	Browse to select an HTML or text file no larger than 666K.
Description	A description for the packet of news articles.

5. Select [Send](#). The selected user receives the notification.

1.5.5.5 Managing and Monitoring Applications

Use Mobile Services cockpit to manage applications, registrations, users, back-end connections to the data source; view application usage statistics; and manage and view application reports.

In Mobile Services cockpit, you can view the properties of Fiori applications and connections that were developed using other tools and imported into SAP Mobile Services, but you cannot edit their properties; input fields and buttons are disabled or hidden.

[Managing Applications \[page 210\]](#)

Manage applications from a single location.

[Managing a Single Application \[page 224\]](#)

A single-app version of Mobile Services cockpit is available for Cloud Foundry. This enables a Business User to manage a single Cloud Foundry application, for example, within a Software as a Service (SAAS) context. The scope of single-app support includes Native, Hybrid, Micro App, and SAP Mobile Cards.

[Exporting and Importing App Configurations \[page 226\]](#)

The ability to import and export application configurations enables you to copy a configuration from one environment to another. For example, you can export an app configuration from the testing environment, and import it in the production environment.

[Exporting App Configurations using SAP Content Agent Service \[page 229\]](#)

The SAP Content Agent service integration allows you to export all or parts of the mobile app configuration.

[Managing Custom Domains \[page 232\]](#)

Manage a custom domain that is associated with a mobile application in the Cloud Foundry environment.

[Managing Push Notifications \[page 233\]](#)

Manage push notifications for the selected application.

[Configuring SAP Cloud ALM \[page 235\]](#)

Configure the integration between SAP Mobile Services and SAP Cloud ALM. This integration enables you to use SAP Cloud ALM to manage the application lifecycle for Mobile Services apps.

[Application Logs and Trace Files \[page 236\]](#)

Set the verbosity for application and component logging, and define how long to keep logs and trace files. You can view all application logs or a subset of your choice, and drill down to view detailed log and trace information if available.

[Analytic Charts \[page 242\]](#)

View analytic charts for the Cloud Foundry environment.

[Managing Alerts \[page 258\]](#)

You can configure alert notifications for SAP BTP resources and Mobile Services using the SAP Alert Notification service. You can also subscribe to event and crash logs from Mobile Services cockpit and configure alert notifications in the Alert Notification service.

[Managing Application Themes \[page 267\]](#)

(Native/MDK only) As an administrator you can manage application themes from Mobile Services cockpit and indicate a default theme for mobile apps.

[Managing App Catalog Settings \[page 273\]](#)

As an administrator, you can manage app catalog settings.

[Managing Cloud Build Service Settings \[page 277\]](#)

(Not available in all COUNTRY/REGIONS) As an administrator, you can manage cloud build signing profiles, and purge cloud build artifacts at the tenant level.

1.5.5.5.1 Managing Applications

Manage applications from a single location.

Context

You can add, edit, or delete applications, and export and import them. You can view all applications, or retrieve a filtered subset of applications; you can sort by column, and refresh the list. From the list of applications, you can drill down to see summary and detailed information for a particular application. You can lock an application to prevent accidental changes. If an app is locked, you must unlock it to modify it.

For apps that have been migrated to single-service apps, enable detailed event logging at the application level. You can select individual services to include, and Mobile AppRouter, as described for the [Log Settings](#) tab below.

For multiple-services apps (those that have not been migrated to single service), you must enable detailed event logging at the service level. Select [Enable Detailed Event Log](#) to collect detailed messages. Select [Disable Detailed Event Log](#) to stop collecting detailed messages (error messages are still collected).

Access ► [Analytics](#) ► [Logs](#) ► to view the event logs, as described in *Viewing Event Logs*.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ► or [SAP Mobile Cards](#).

You can view a list of applications with summary information such as name, application ID, vendor, application state, creation date, and so forth. Use the search and sort options to locate the applications you're interested in.

Note

Apps managed by App Lab do not appear in the application list. To manage App Lab updates, see *Managing Application Versions using App Lab*.

Native/MDK Application Summary

Column	Description
Application ID	The unique application identifier.
Name	The application name.
Vendor	The vendor who supplied the application.
License Type	The application plan that was in place when the application was created, for example, "resources", "build-code", "free", "standard (Users)", and "b2c (Consumer Edition)". The "kernel-service" plan is used for an application that is managed in a single-app cockpit (Software as a Service, SaaS). The "lite" plan is available only for trial subaccounts. The license determines capabilities and charges. See <i>Service Plans Changing Service Plans at the App Level</i> for additional information.
State	The current state of the application such as: <ul style="list-style-type: none">• Started – the application is running normally.• Inconsistent – applications may become inconsistent if they are deleted or changed from the SAP BTP cockpit. From Mobile Services cockpit, you can either delete or restore the application.• Deleted from SAP BTP cockpit – the application was deleted. You can restore or delete it in the Mobile Services cockpit.• Restoring – the application is being restored from an inconsistent state.

Column	Description
Outdated	Whether the application is outdated. If checked, you are prompted to update the application to a newer version that contains fixes and improvements. Typically you should update the application.
Creation Date (UTC/local)	The application creation date, either in UTC or local format, depending on what you selected for your User Setting.

2. Select an application to view its details. To prevent accidental changes, you can lock an application once its configurations are complete. To make changes, you'll need to manually unlock the application,
 - [Info](#) provides a summary of settings for the selected application. To set the information on this tab, see *Defining Applications*. In the Cloud Foundry environment you can also configure a [Custom Domain](#). See *Editing an Application* and *Managing Custom Domains*.
The list under [Assigned Features](#) identifies the features that are currently assigned to the application, and enables you to add and delete more. For more information, see *Configuring Assigned Features*.
 - [APIs](#) provides onboarding and API features.
 - Under [Onboarding](#), you can find one or more QR codes that can be used to configure the mobile application to connect to this Mobile Services instance.
For [In-app Scanning Code](#), the device user must first manually launch the app, then scan the QR code from within the app to initiate the onboarding process.
For [Default Configuration](#), native app developers can download a native-specific file containing the Discovery Service default configuration. Select [iOS](#) to download the configuration file in `plist` format, or select [Android](#) to download the file in `JSON` format. Simply add the file to your iOS or Android app project without further formatting.
For [Configuration](#), you can view any application routes that you've associated with the application. Select edit to make changes. See *Defining Applications (Cloud Foundry)* for information about the XSUAA property, [Domain of Application Route](#).
For [Apple and Android Launch App Code](#), the device user can select the URL (or scan the QR code for Apple's Camera App or an Android application that allows reading QR codes) to find and launch the application installed locally. If the user enables Android app links, it also supports Android's camera app. The App-launch QR code is generated if you enable Apple Universal Links or Android App Links or both of them in the [Application Links](#) section; the label text of the QR code depends on which application links are enabled ("Apple launch app code," "Android launch app code," or "Apple and Android launch app code."
For [Apple and Android Launch App Code with URI Scheme](#), the device user can scan the QR code or select the URL, to use an appLink URL to find and launch the application. This option only appears if you enabled Device Application URI Schemes in the [Application Links](#) tab. You must select one Device Type, and input the corresponding device application URI Scheme for iOS or Android.

📌 Note

(Mobile Development Kit only) If you set up Apple and Android Device Application URI schemes in the [Application Links](#) tab, you see two separate QR codes, labeled [Apple Launch App Code with URI Scheme](#) and [Android Launch App Code with URI Scheme](#)

If the app security configuration changes, you must generate a new QR code. Also, please note that the URL can be very long and might be longer than the value shown on the screen. Use [Copy to Clipboard](#) (📋), to copy its complete value.

Under [API](#), you can view a list of frequently used API URLs. Use [Copy to Clipboard](#) (📋), to copy read-only URLs to the clipboard, to paste elsewhere, avoiding typing errors and ensuring accuracy.

📘 Note

The Server URL is also the `<mobile services host>` name.

- You can use [Application Links](#) to enable universal links for Apple and application links for Android apps. Apple universal links and Android application links enable users to open an application locally without downloading the app or using a browser container.
- [Users](#) enables you to export or delete data for a user, and to block one or more users from using a Native/MDK or SAP Mobile Cards application.
- [Security](#) enables you to manage security for the selected application. For details see *Configuring App Security*.
- [Alert](#) lets you configure settings for server failures in various categories. Once configured, implement the SAP Alert Notification Service to receive notifications. See *Configuring Alert Settings* and *Subscribe to Mobile Service Alerts* for information.
- [Log Settings](#) is for apps that have been migrated to the single-service model, and is used to enable event logs at the application level. You can specify any of the individual services that are assigned to the app, as well as [Mobile AppRouter](#). This gives you more control over the event log information you want to see at the application level. See *Enabling Event Logs for Apps*. You can continue to view event logs as described in *Viewing Event Logs*.

[Editing an Application \[page 214\]](#)

Edit an existing application from the application list.

[Deleting an Application \[page 214\]](#)

Delete an application from the application list. For any SAP Fiori or hybrid apps regenerated with the cloud build service, deleting the application also permanently deletes any binaries that are associated with the application.

[Changing Service Plans at the App Level \[page 215\]](#)

If you change Service Plans at the SAP BTP level, you may also need to change service plan settings at the Mobile Services app level.

[Creating Application Links \[page 216\]](#)

Enable application links for apps. Once enabled and configured, a QR code is generated.

[Managing App Users \[page 218\]](#)

Manage users at the application level.

[Managing Registered Users \[page 219\]](#)

Manage user registrations for the selected app from the Mobile Settings Exchange feature.

[Configuring Alert Settings \[page 222\]](#)

Administrators can configure alert settings for server-side failures for the selected application.

[Enabling Event Logs for Apps \[page 223\]](#)

For apps that have been migrated to the single-service model, you can enable detailed event logging at the application level.


Related Information

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)
[Configuring Assigned Features \[page 53\]](#)
[Managing Application Versions Using App Lab \[page 175\]](#)
[Editing an Application \[page 214\]](#)
[Managing Custom Domains \[page 232\]](#)
[Creating Application Links \[page 216\]](#)
[Service Plans \[page 23\]](#)

1.5.5.5.1.1 Editing an Application

Edit an existing application from the application list.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ►
2. Select an application, and click  on any tab to make modifications. See *Defining Applications* for application properties.
3. Save your changes.

Related Information

[Managing Custom Domains \[page 232\]](#)

1.5.5.5.1.2 Deleting an Application

Delete an application from the application list. For any SAP Fiori or hybrid apps regenerated with the cloud build service, deleting the application also permanently deletes any binaries that are associated with the application.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ► or *SAP Mobile Cards*.

2. Select an application and click [Delete](#).
3. Click [OK](#) to confirm.

Note

Once an application has been deleted, users won't be able to use it. All existing logs and traces are deleted and cannot be retrieved. For SAP Mobile Cards, the card template is deleted, not the cards.

1.5.5.5.1.3 Changing Service Plans at the App Level

If you change Service Plans at the SAP BTP level, you may also need to change service plan settings at the Mobile Services app level.

Context

The terms license type and service plan are used interchangeably. The license determines capabilities and charges.

Before you start, it is recommended that you have a good understanding of the service plans available, and your current service plan. You can check the SAP BTP cockpit to see your current active Mobile Services service plans. Be aware of whether the Mobile Services app and Cloud Foundry Runtime are under the same Service Plans. To be billed properly, all existing services must be on the right service. For example, if the app is under the "free" plan, the Cloud Foundry Runtime must also be under the "free" plan or charges are incurred.


For more information, see:

- *Service Plans*
 - *Important Free Plan Information*
 - *Upgrading the Cloud Foundry Runtime*

Also keep in mind these guidelines:

- The "standard (Users)" and "b2c (Consumer Edition)" service plans are deprecated and are not a good long-term selection.
- You can change your service plan from "free" to "resources" or "build-code", enabling you to migrate your app without losing your work.
- Migration is not reversible. You cannot switch from "resources", "build-code", "standard (Users)", or "b2c (Consumer Edition)" to "free".
- The "kernel-service" license type is used for an application that is built by SAP and managed in a single-app cockpit (Software as a Service, SaaS). The license type is defined by SAP and thus, is read-only and cannot be selected or edited in the cockpit.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, and in the **Application Details** section, click  to make modifications.
3. In **Edit Application Details**, locate the **License Type** field. Its value represents the license type that was in place at the time the app was created or deployed in Mobile Services cockpit. For example, you may have started with a "free" plan (non-trial landscape) or a "lite" plan (trial subaccounts).
4. Select the target license from the **License Type** drop-down list, keeping in mind guidelines and limitations. For example, if you currently have the "free" plan, you could select "resources" to change plans at the app level.
5. Save your changes.

It is a good idea to monitor your billing to make sure it reflects the changes you intended to make.

Related Information

[Service Plans \[page 23\]](#)

1.5.5.5.1.4 Creating Application Links

Enable application links for apps. Once enabled and configured, a QR code is generated.

Context

These application link and QR code options are available:

- In-apps QR code – scan the QR code from within the device application.
- Onboarding QR code – scan the QR code to also launch the device application, and onboard users. The onboarding QR code includes the device application app_id, which the device uses to find and launch the local installation of the application.
- Application link QR code – scan the QR code to launch the device application, and onboard users. The onboarding QR code includes an appLink URL that is supplied by mobile services. The device sends a request to the appLink URL and retrieves information about the device application. Then, the device uses this information to find and launch the application.

You can configure Apple universal links and Android application links to enable users to open an application locally without downloading the app or using a browser container. You can also configure Apple and Android URI links for third-party applications and devices.

Procedure

1. In Mobile Services cockpit, select ► *Mobile Applications* ► *Native/MDK* ►, and then an application
2. Select *Application Links*.
3. Under *Apple Universal Links*, create or modify a universal link.

Apple Universal Link Settings

Setting	Description
Enabled	Whether universal links are enabled for the app on iOS
Team ID	The team or app identifier associated with the app
Bundle ID	The resource bundle identifier associated with the app

For more information, see: <https://developer.apple.com/library/content/documentation/General/Conceptual/AppSearch/UniversalLinks.html> ►.

4. Under *Android App Links*, create or modify an application link.

Android Application Link Settings

Setting	Description
Enabled	Whether application links are enabled for the app on Android
Site Domain	The URL for the site domain where the app is hosted, in the format <code>https://mobile-app-router-mobile-app.cf.sap.ondemand.com</code>
Association File Content	The content of <code>assetlinks.json</code> , which indicates the Android apps that are associated with the current website and verifies the app's URL intents.

For more information, see: <https://developer.android.com/training/app-links/index.html> ►.

5. Under *Device Application URI Schemes*, create a third-party application link based on an application URI or device application type. This form only appears if application migration status is complete.

Device Application URI Settings

Setting	Description
Apple Device Application URI Scheme	Enter a name for the Apple device launch code. This name will be incorporated into the launch app code with URI scheme link.

Setting	Description
Android Device Application URI Scheme	Enter a name for the Android device launch code. This name will be incorporated into the launch app code with URI scheme link.
Device Application Type	Select <i>Generic</i> or <i>Mobile Development Kit</i> .

Note

Mobile Development Kit only supports applications using OAuth security, otherwise the QR code and URI will be missing.

Next Steps

Once the universal or application link has been enabled, the QR code and URL appear on the *APIs* tab. Depending on your entries, you may see one or more of the following:

- In-app Scanning Code (with default configuration downloads for iOS and Android)
- Apple Launch App Code with Universal Link
- Android Launch App Code with App Link
- Apple Launch App Code with URI Scheme
- Android Launch App Code with URI Scheme

You can distribute the QR code, and users can launch the application locally without downloading it and without using a browser container. See *Managing Applications*.

Related Information

[Managing Applications \[page 210\]](#)





1.5.5.5.1.5 Managing App Users

Manage users at the application level.

Context

You can delete or export data for a selected user, and block users.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application and then [Users](#).
3. (Optional) Under [User Data](#) you can export or delete data for an individual user.
 1. Enter the user name. Valid entries include alphanumeric characters, numbers, period, underscore, and hyphen.
 2. (Optional) Select [Export](#) to download user data to a JSON file.
 3. (Optional) Select [Delete](#), to delete user data, and select [OK](#). The message `Delete user data successfully` indicates success.
4. (Optional) Under [Blocked Users](#), you can block one or more users from using the selected Native/MDK or SAP Mobile Cards application.
 1. Select .
 2. Add the user you want to block.
 3. Click [OK](#) to confirm.

You can also block users through the Mobile Settings Exchange feature as described in *Managing Registered Users*.




1.5.5.5.1.6 Managing Registered Users

Manage user registrations for the selected app from the Mobile Settings Exchange feature.

Context

Mobiles Services uses the Mobile Settings Exchange feature to exchange general settings and user registration information between mobile client and server. Use this information to manage user registrations.

Procedure



1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#)  or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#)
3. Select [User Registrations](#) to manage registered application users.
4. Under [Automatic Removal](#), select [Automatic Removal](#) to indicate whether application registrations should be automatically removed after a specified number of days.
5. Under [Summary](#), view the number of current application registrations.
6. Enter search criteria to narrow the focus to specific users or groups of users, and then select [Go](#). You can search based on [User Name](#), [Registration Time Frame](#) (either a predefined time frame, or a custom time

frame using the calendar date-picker to define a range), [Device Type](#), [Email Address](#), [Compliance Status](#) (Compliant, Compromised, or Unknown), or [Wipe Status](#) (not wiped or wiped).


7. You can view the search results to see registered users.

Registered Users

Column	Description
Registration ID	Unique identifier provided by the client application, or system-generated application registration ID. A selection box appears to the left of each registered user.
User Name (Blocked in Red)	User of the registered application. Users who are blocked are marked in red.
Device ID	Device identification.
Device Type	Type of device, such as Android or iPhone, sent by the device during registration/onboarding. "Unknown" indicates that the device type cannot be detected.
Last Connection (UTC+0000)	The date, time, and time zone when the application was registered, in the format YYYY-MM-DD HH:mm:ss.
Wipe	Whether the application connection is set to be wiped.
Compliance Status	Whether the Mobile Services server has determined that a client device meets certain security requirements. Options include: <ul style="list-style-type: none"> Compliant - the device has undergone compliance detection, and was found to not be either jailbroken (Android) or rooted (iOS). Compromised - the device has undergone compliance detection, and was found to be either jailbroken (Android) or rooted (iOS). The compromised device status is also reported as an Error in the Event Log (use the Mobile Device Security filter). Once the compliance problem is fixed and rechecked, the status changes. Unknown - the device has not undergone compliance detection, so its status is not yet known.
Last Revocation (UTC+0000)	The date, time, and time zone when the OAuth token was last revoked in the format YYYY-MM-dd HH:mm:ss. The date and time stamp is updated when you click the Revoke button. The time stamp is also updated if the user logs out from a device. When the user logs out from a device, all tokens under the same user name are revoked.
Actions	Actions to take, if any. Actions include sending a push message to a registered user, locking or unlocking a user, blocking and wiping a user, deleting a user registration, and revoking an OAuth token.


- Send a Push Notification to a User
Select  to send a message to a user. In [Send Push Notification](#), type your message and then click [Send](#). The message The native push notification was sent successfully confirms that the message was sent.
- Block a User
Select  to block a user. Blocking prevents a user from registering the Native/MDK application or receiving traffic. The [Confirm Block User Action](#) message appears, informing you that by blocking user access, the user is logged out of the application and cannot log back into the application unless


removed from the list of blocked users. Select [OK](#) to confirm. The message `User Blocked` appears and a checkmark appears to the right of the icon , indicating the user is blocked.

Select the icon to unblock the user. The [Confirm Unblock User Action](#) message appears, informing you that by unblocking user access, the user regains access to the application and can log in through any device. Select [OK](#) to confirm. The message `User Unblocked` appears and an X appears to the right of the icon , indicating the user is unblocked.

See *Managing App Users* for related information.


- Lock User and Wipe Device


The “Lock User and Wipe Device” button  is designed to enhance the security and management of registered users and their devices in the system. When this button is clicked, the system logs out the selected users from the application on the specified devices and wipes all data managed by the app on those devices. This action is irreversible, meaning that any previously wiped data managed by the app is not restored.

The “Unlock User” button  allows administrators to regain access for a user on a specific device after they have been locked out and their data has been wiped. This button disables the “Wipe” function for the selected registration and unlocks the user, allowing them to log back into the application on the same device. However, it is important to note that any previously wiped data managed by the app is not restored.

See *Defining Lock and Wipe Policy* for related information.

- Delete a User Registration

Select  to delete an individual registered user. You can also use the check box to the left of the


Registration ID, to select one or more rows, and then the  button that is on the Registered Users row (not the delete button for individual user rows). Select [OK](#) to confirm. The selected registered users are removed.

For apps configured for OAuth security, when you delete a user registration, all related OAuth tokens are automatically revoked. This is the case whether the app calls the Admin API or the Client Runtime API.

- If the user has only one registration in the list: when it's deleted, the user registration is revoked and the user registration no longer appears in the list.
- If the user has multiple registrations in the list: when one is deleted, the other user registrations are also revoked and the user registrations no longer appear in the list.

In either case, when the user registration is deleted, old tokens are deleted, and user registrations do not appear in the list and are not included in metering data. When the user logs back in to the app, a new OAuth token is automatically generated. The user registration again appears in the list and is counted when sending metering data. This feature helps ensure that only the latest OAuth tokens are used and that old tokens are deleted.

Note

For some push providers, when you delete user registrations for auto registration in [Mobile Settings Exchange](#), you must also manually delete the device registration from [Mobile Push Notifications > Push Notifications](#) . This is because the provider does not provide the info needed for auto deletion in push. For these providers, you must perform the manual delete: Baidu, Custom, and W3C.

- Revoke an OAuth Token

Select [Revoke](#) to revoke all OAuth tokens used by a registered user. When the user next logs in, a new OAuth token is issued, ensuring that the most current OAuth token is used. This feature is available for applications that use the OAuth Security configuration, including apps using Identity Authentication service (IAS) and Software as a Service (SaaS) apps. See *Revoking an OAuth Token* for information. If the [Revoke](#) button does not appear, the feature is not supported for the configuration.

Related Information

[Revoking OAuth Tokens \[page 301\]](#)

1.5.5.5.17 Configuring Alert Settings

Administrators can configure alert settings for server-side failures for the selected application.

Prerequisites





The SAP Alert Notification service must be available for the Cloud Foundry environment in order to receive alerts. If the Alert Notification service is not available, this message appears on the [Alerts](#) tab: `This feature requires Alert Notification Service. Alert Notification Service is not yet available in this landscape or region.`


Context

Once the Alert Notification service is available, an administrator enables alerts for the application, and establishes the message threshold, and schedule.

To receive notifications, configure Subscriptions and Alerts through the Alert Notification service (subscriptions and alerts cannot be managed through Mobile Services cockpit). See *Subscribe to Mobile Service Alerts* for information.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, then select the [Alert](#) tab.
3. Select .
4. In [Edit Settings](#), select [Enable Alert](#).

- Message Threshold – if the generated error/log number is greater than the threshold number send an alert notification.
 - Schedule – frequency at which alerts are sent.
 - Expires – the date until which alerts are sent. This value is not editable, and changes automatically when the administrator makes any changes to this page. Alerts are sent only for 60 days after the change.
5. Under [Services Alert](#), select  to set the event log level to Error or Warning for each service.

Note

- All features assigned to the selected app are listed here.
- Mobile Sample OData ESPM feature is not supported.

6. In [Alert Notification Service](#), you are referred to the Alert Notification Service available in the SAP BTP Cloud Foundry environment. You can configure alerts for one or more services. See *Subscribe to Mobile Service Alerts* for information.
7. Click [OK](#).

Related Information

[Subscribe to Mobile Service Alerts \[page 258\]](#)

1.5.5.5.1.8 Enabling Event Logs for Apps

For apps that have been migrated to the single-service model, you can enable detailed event logging at the application level.

Context

You can specify the services to include in the detailed event log, as well as [Mobile Application](#). This gives you more control over the event log information you want to see at the app level.

Note

Detailed logging is used for troubleshooting, not for monitoring, so logging is terminated automatically after 24 hours. You can manually enable detailed event logging for another 24 hours.


For applications that have not been migrated to the single-service model, you must enable detailed event logs at the service level.

You still view the event logs for the application as described in *Viewing Event Logs*.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#) (or [Mobile Applications](#) > [SAP Mobile Cards](#) or [Mobile Applications](#) > [Micro App](#)).
2. Select an application, then select the [Log Settings](#) tab. Under [Detailed Event Log Settings](#), all features that are assigned to the application appear. (If [Log Settings](#) does not appear, it means the app has not been migrated to the single service model).

Settings	Description
Feature Name	All features that are assigned to the application appear, as well as the additional feature, Mobile Application .
Included in Detailed Event Log	Whether the feature is included in the detailed event log for the selected application.

3. Select .
4. In [Enable Detailed Event Log Settings](#), select the features to include in the detailed event log for the application. Only features assigned to the app are available, plus [Mobile Application](#).
5. Click [Enable](#). Changes appear under [Included in Detailed Event Log](#).

1.5.5.5.2 Managing a Single Application

A single-app version of Mobile Services cockpit is available for Cloud Foundry. This enables a Business User to manage a single Cloud Foundry application, for example, within a Software as a Service (SAAS) context. The scope of single-app support includes Native, Hybrid, Micro App, and SAP Mobile Cards.

Prerequisites

The feature is only available for Cloud Foundry, and requires that the application has been migrated to a single service.

The following roles are related to the Business User cockpit. To learn more about setting roles and role collections see [Building Roles and Role Collections for Applications](#).

- **Administrator** – read-write role. The Business User must be assigned the Administrator role.
- **Helpdesk** – read-only role.

Access to the single-app Mobile Services cockpit is via a specific cockpit URL. The URL is in the format `https://<mobile_application_url>/mobileservices/Admin`. For example, if the URL for the `com.sap.myApp` application is `http://mobile-tenant1-groupX-tenant-com-sap-myapp.cfapps.sap.hana.ondemand.com`, then the application is accessible at: `https://mobile-tenant1-groupX-tenant-com-sap-myapp.cfapps.sap.hana.ondemand.com/mobileservices/Admin`

→ Tip

You can find the `<mobile_application_url>` by navigating to the [APIs](#) tab, and copying the server URL.

Context

Regular Mobile Services cockpit access is only possible as a Platform User. The Platform User can also log in to the SAP BTP cockpit, and is a member of the sub-account, organization and space.

The Business User, on the other hand, is authenticated against the Trust provider that is configured in the SAP BTP sub-account. The Business User does not require general access to SAP BTP accounts.

The functionality of the Business User is restricted. Because the logged in user is not allowed to perform actions on SAP BTP accounts, Mobile Services cannot run operations that require interaction with the Cloud Foundry infrastructure and other services. For example, the Business User cannot create a new application, which installs a new application route in the space; or change security settings, which would re-create the XSUAA (Security) Service Instance.

You'll notice these differences when using Mobile Services cockpit as a Business User:

- The left navigation pane no longer appears, and the navigation breadcrumbs are abridged, since there is only one app to manage.
- The [Delete](#), [Export](#), and [Lock](#) operations no longer appear.
- Several assigned features no longer appear, including [Mobile App Catalog](#) and [Mobile Cloud Build](#).
- The [Discovery](#) tab no longer appears.
- On the [Info](#) tab under [Application Details](#), [Custom Domain](#) is read only; and the [License Type](#) either does not appear or appears as "kernel-service", depending on the app.
- On the [Security](#) tab, the [OAuth Settings](#) and [Role Settings](#) are read only.
- On the [Logs](#) tab, you can view event logs only for the single application.

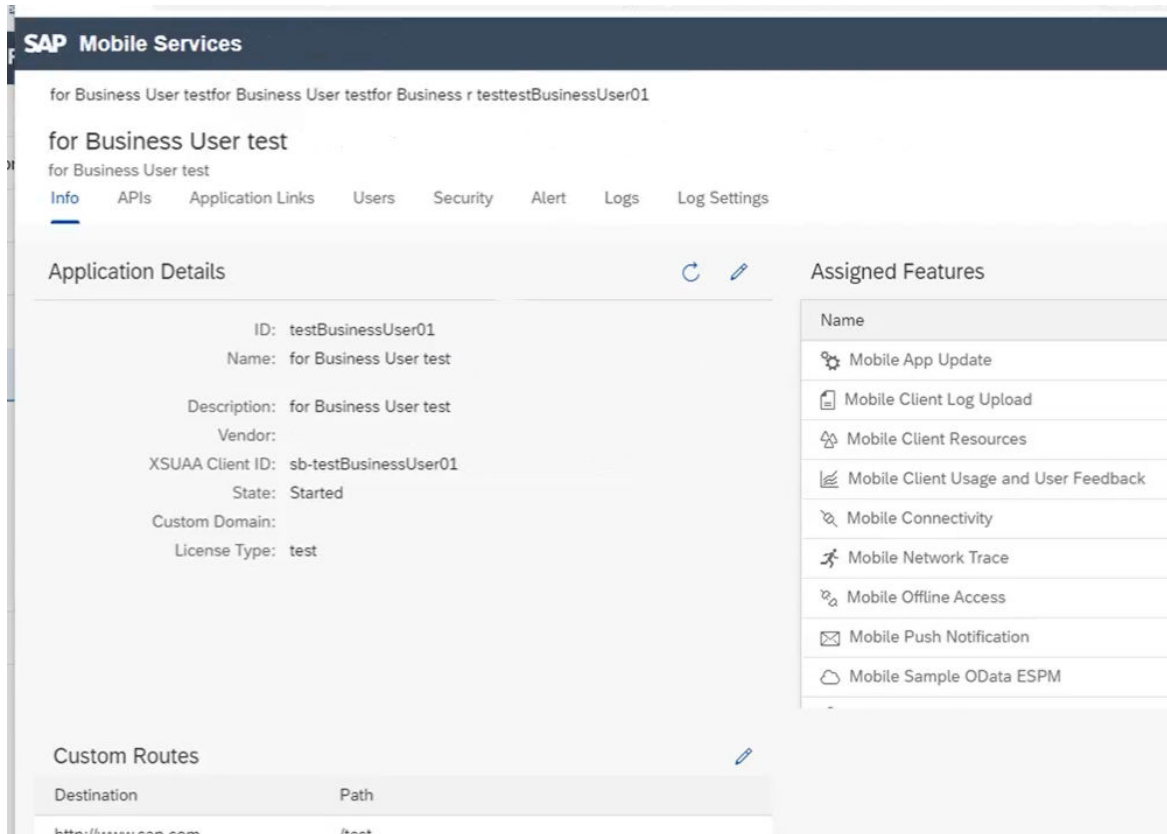
Other than these basic differences, most Mobile Services cockpit operations are the same for the single-app version.

Procedure

1. Access the single-app Mobile Services cockpit from a browser using the cockpit URL in the format:
`https://<mobile_application_url>/mobileservices/Admin`, as described above in [Context](#).

For example, `https://mobile-tenant1-groupX-tenant-com-sap-myapp.cfapps.sap.hana.ondemand.com/mobileservices/Admin`. You are prompted to log in, and then Mobile Services cockpit appears with the single application.

2. Use the Mobile Services cockpit to manage the application and its assigned features, keeping in mind the differences and restrictions mentioned above in [Context](#).



Related Information

[Viewing Event Logs \[page 237\]](#)

1.5.5.5.3 Exporting and Importing App Configurations

The ability to import and export application configurations enables you to copy a configuration from one environment to another. For example, you can export an app configuration from the testing environment, and import it in the production environment.

Export/Import Application Options

SAP Mobile Services offers you two options to move app configurations from one environment to another:

- **Using SAP Mobile Services export/import features**
This option copies the entire mobile app configuration from one environment to another. Existing mobile apps cannot be overridden and must be deleted before a new configuration is imported. Sensitive information, like passwords and keys, are excluded from the export.

- **Using the SAP Content Agent service export feature**

This option allows you to export all or parts of the mobile app configuration. The SAP Content Agent service initiated import creates the mobile app configuration on the target, creates missing features, and merges the exported configuration, as described in [SAP Content Agent Service User Guide](#).

[Exporting Application Configurations \[page 227\]](#)

Export an application configuration `zip` file to your local system. You can use the export feature to create a back-up of the application configuration, and as a prerequisite for importing the application configuration to SAP Mobile Services.

[Importing Application Configurations \[page 228\]](#)

Import an application configuration from one SAP Mobile Services environment to another.

Related Information

[Exporting App Configurations using SAP Content Agent Service \[page 229\]](#)

1.5.5.5.3.1 Exporting Application Configurations

Export an application configuration `zip` file to your local system. You can use the export feature to create a back-up of the application configuration, and as a prerequisite for importing the application configuration to SAP Mobile Services.

Prerequisites

Before exporting an application to your local system, verify that the application status is *consistent* (marked in green).

Context

Only the current version of an application is exported; *new* versions are not exported.

The `.zip` file contains multiple files, one for the application and one for each service. Basic configuration is included, such as OAuth settings, and configuration from each service instance connected to the application.

Note

You can also export configuration data for individual assigned features, as described in *Exporting Data*.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select the application, and then select [Export](#) in the upper right corner.

The application configuration file is downloaded to the `<appname>.zip` file in the default location, which is specified in the browser.

Note

You can export the application configuration to a shared directory to make it available to all the mobile services systems that have access to the shared directory.

1.5.5.5.3.2 Importing Application Configurations

Import an application configuration from one SAP Mobile Services environment to another.

Prerequisites

Verify that the application `*.zip` file you want to import is available on the SAP Mobile Services network. You must export the application from SAP Mobile Services (Neo) or SAP Mobile Services (Cloud Foundry) before you can import it into your Cloud Foundry environment.

Context

You can import mobile app configurations from SAP Mobile Services Neo or SAP Mobile Services Cloud Foundry landscapes to Cloud Foundry. Most configuration settings are preserved, and a few must be reentered for the new environment.

Note

You can import a Fiori application that was developed with other tools, but you cannot edit its properties in Mobile Services cockpit.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select [Import](#) in the upper right corner.

3. Click [Browse](#), and select the application configuration *.zip file that you exported from SAP Mobile Services Neo or Cloud Foundry.
4. Click [Save](#).

When the import completes, you see a message like `Application imported successfully`.

If you import an application with the same ID as an existing application, you see the error message `Application already exists`.

Next Steps

Configure the application for the target server or Cloud Foundry environment. Keep in mind these guidelines:

- Passwords—such as those for APNs, security, and proxy endpoint SSO—are not imported. The value is set to asterisks to protect security. You must manually reset them.
- Some Neo application features can't be imported to Cloud Foundry. For example:
 - The Document Repository, JSON Storage and Alert features are not supported in Cloud Foundry.
 - The Client-resource, App-Update, and Cloud Build feature configurations can't be imported into Cloud Foundry.
- When importing the Neo application Connectivity feature, some endpoint SSO types must be checked and reset:
 1. For the AppToAppSSO SSO type, Cloud Foundry sets mock value asterisks for some required properties. You must manually reset the mock values to the correct values, if the [Issuer](#) and [Audience](#) property values appear as asterisks. The [Signing Key](#) property value must also be reset.
 2. For the OAuth2SAMLBearerAssertion SSO type, Cloud Foundry sets mock value asterisks for some required properties. You must manually reset the mock values to the correct value, if the [SAML Assertion Issuer](#) property value appears as asterisks. The [Client Secret](#) and [Signing Key](#) property values must also be reset.

Related Information

[Migrate: SAP Mobile Services Neo to Cloud Foundry \[page 530\]](#)

[Get Started \[page 18\]](#)

1.5.5.5.4 Exporting App Configurations using SAP Content Agent Service

The SAP Content Agent service integration allows you to export all or parts of the mobile app configuration.

The SAP Content Agent service initiated import creates the mobile app configuration on the target, creates missing features, and merges the exported configuration. SAP Content Agent service integrates into SAP Cloud Transport Management too, allowing transport of applications. You first need to subscribe to the SAP

Content Agent service on the source and target spaces. For details, see [SAP Content Agent service initial setup](#).

The components provided by SAP Mobile Services are listed on this page. The configuration of mandatory components are always exported. The components are grouped by feature.

Application Base Configuration Options

Component	Description	Location	Export	Import Behavior
Application Base Information	The base application information, such as Name, ID, Vendor, and so forth	Info tab	Always	Configuration is overridden by import
Application Links	Application links enable you to seamlessly deep link from web pages to app content	Application Link tab, see Application Links	Optional	Configuration is overridden by import
Route Configuration	Application Route Configuration	Info tab	Optional	
Custom Routes	Custom Routes	Info tab	Optional	Configuration is overridden by import
Versioning	Restrict access to defined set of application version	Security tab, see Application Versioning	Optional	Configuration is overridden by import
Security Configuration	Configures the application security	Security tab, see Security Versioning	Optional	Configuration is overridden by import
XSUAA Configuration	Configures the XSUAA settings	Security tab, see XSUAA Settings	Optional	Configuration is overridden by import

Features

Each mobile app export includes the list of activated features, except Mobile Sample OData ESPM, Mobile Transaction Build, Mobile App Catalog, Mobile Cloud Build and Mobile Network Trace. All exported features are created in the target instance.

Component	Feature Name	Description	Export	Import Behavior
App Update Current Revision	Mobile App Update	Staged or active applications	Optional	Import overrides existing configuration
Application configuration	Mobile Settings Exchange	Password Policies and Features	Optional	Password policy is overridden by import. New features are added. Existing

Component	Feature Name	Description	Export	Import Behavior
				features aren't overridden
Augmented Reality	Mobile Augmented Reality	Scenes for augmented reality	Optional	Import overrides existing configuration
Client Log Upload	Mobile Client Log Upload	The client log upload feature	No configuration export	Feature is created only
Client Resources	Mobile Client Resources	Client Resources	Optional	Import overrides existing configuration
Proxy	Mobile Connectivity	The mobile destination configuration	Optional, no secrets or credentials are exported	Missing destinations are created, existing destination configurations aren't overridden
Offline	Mobile Offline Access	The mobile offline configuration	Optional	Configuration is overridden by import
Base Configuration	Mobile Push Notifications	Base notification configuration: Predefined Push Configuration and Push User Name	Mandatory	Import overrides existing configuration. Predefined Push Configuration disables all other push provider configuration
Android	Mobile Push Notifications	Android notification configuration	Optional	Import overrides existing configuration
Apple	Mobile Push Notifications	Apple notification configuration	Optional	Import overrides existing configuration
Baidu	Mobile Push Notifications	Baidu notification configuration	Optional	Import overrides existing configuration
Custom	Mobile Push Notifications	Custom notification configuration	Optional	Import overrides existing configuration
W3C	Mobile Push Notifications	W3C notification configuration	Optional	Enables/Disables configuration. A new key is generated when enabled
WNS	Mobile Push Notifications	Windows Notification Service configuration	Optional	Import overrides existing configuration

References

You can find details to SAP Content Agent service and SAP Cloud Transport Management here:

- [SAP Content Agent User Guide](#)
- [Export Content Wizard](#)
- [Import Content Wizard](#)

- [SAP Cloud Service Content Types](#)
- [SAP Cloud Transport Management](#)

1.5.5.5.5 Managing Custom Domains

Manage a custom domain that is associated with a mobile application in the Cloud Foundry environment.

Prerequisites





- You must have a custom domain already configured and available on Cloud Foundry.
- You must know how to use the domain already, before configuring it for a mobile application.
- When you define the mobile application to Mobile Services cockpit, use either SAML or OAuth authentication.

Context

Subaccount owners can make an applications accessible via a custom domain that is different from the default domain. You can configure mobile services apps to use the custom domain. You can configure one custom domain in the Mobile Services cockpit.

Some regions may support multiple domains. If so, the administrator associates an application with the domain of the application route, to ensure the correct domain is used. See *Defining Applications (Cloud Foundry)* for information about the XSUAA property, [Domain of Application Route](#).

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) , and select an application.
2. Once the application is defined, you can associate the custom domain name, such as `cfapps.sap.hana.ondemand.com` with the application. See *Defining Applications (Cloud Foundry)* for information.
3. Once the application is saved, on the [APIs](#) tab, you can select the  button to edit the domain configuration.
4. On [Info](#) you can check [Custom Domains](#) to view the current custom domain.

1.5.5.5.6 Managing Push Notifications

Manage push notifications for the selected application.

Context


This feature uses the existing push notification mechanism to send messages to recipients. You can send a native push notification. You must be assigned the Developer or Manager role for the space to send push notifications (a disabled [Send Push Notification](#) button indicates the role needs to be added to your profile).

Note

For some push providers, if you deleted user registrations for auto registration in [Mobile Settings Exchange](#), you must also manually delete the device registration from [Mobile Push Notifications](#) [Push Notifications](#). This is because the provider does not provide the information needed for automatic deletion in push. For these providers, you must perform the manual delete: Baidu, Custom, and W3C.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#) or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select [Features](#) [Push Notification](#).
3. Select [Push Registrations](#) to see a list of registered users who are eligible for push notifications.

You can click the Customize Table Columns icon  to change the columns that appear.

Registered Users Default Columns


Category	Description
User Name	User name of the registered user.
Device ID	The device type for the registered user, such as Android or IOS.
Push Provider	The push provider associated with the device.
Push Group	The push group associated with the device.
Created At (UTC-700)	The time the device registered in UTC format.

Category	Description
Actions	Indicates any actions you can take for the registered user, such as sending a notification, or deleting the registration.

- You can use the filters to help find a subset of registered users.

Filter Criteria

Filter	Description
User Name	Enter a specific user name, or the start of a user name.
User Locale	Enter the language. This field is not validated, so you can enter any string value. Use the field if you know one or more values that users entered during registration, such as EN or German.
Time Zone	Enter one or more time zones, separated by commas. This field is not validated, so you can enter any string value. Use the field if you know one or more values that users entered during registration, such as UTC-1 or PST.
Device Model	Select one or more device model, such as iOS and Android.
Push Group	Enter a push group name. This field is not validated, so you can enter any string value. Use the field to create an ad hoc group for filtering. Push groups do not persist.
Registration Time Frame	Select a time frame from the list: Last 24 Hours, Last 7 Days, Last 4 Weeks, Last 3 Months, Last 6 Months, Last 12 Months, or Custom Defined (use the date-time picker to define a range).

- Select [Send Notification](#) to send a push notification to a registered user.
- In the [Send Push Notification](#) dialog, create your push notification. You can:
 - Select [General](#) and type a push notification message.
 - Select [Advanced](#) and modify the JSON format payload version of the default push notification message. You should only do this if you are an expert user. Select [Revert to Default](#) to revert to the default JSON code.
- Select [Send](#). The native push notification was sent successfully. appears.
- Select  to remove a registered user from the push notification list.

As described in Context above, deleting a registered user might be necessary if you deleted a user registrations for auto registration in [Mobile Settings Exchange](#) for certain vendors, including Baidu, Custom, and W3C.

Related Information

[Enabling Push for Subscribers \(SaaS\) \[page 152\]](#)

1.5.5.5.7 Configuring SAP Cloud ALM

Configure the integration between SAP Mobile Services and SAP Cloud ALM. This integration enables you to use SAP Cloud ALM to manage the application lifecycle for Mobile Services apps.

Prerequisites

Tenants must be registered with SAP Cloud ALM as described in the [Setup for SAP Mobile Services](#) documentation. You must download the [SAP Cloud ALM Service Key](#) to connect to the SAP Cloud ALM system. You must be assigned the Administrator role to configure the integration between SAP Mobile Services and SAP Cloud ALM.

Context

SAP Cloud ALM is a solution tool for application lifecycle management. Mobile Services must be registered to the SAP Cloud ALM server to use the service.

Procedure

1. In Mobile Services cockpit, select **Settings** > **SAP Cloud ALM**.
2. Under **Integration Monitoring** enter SAP Cloud ALM integration values. Obtain the values from the downloaded SAP Cloud ALM service key file.

SAP Cloud ALM integration properties

Property	Description
ALM Service URL	The ALM server base URL. Use the service key parameter "Api", and omit the "/api" portion. For example: <code>https://euXX.alm.cloud.sap.</code>

Property	Description
ALM Authentication Endpoint	The OAuth endpoint URL used for authentication. Use the service key parameter "url", and add /oauth/token. For example: https://calm-tenant.authentication.eu10.hana.ondemand.com/oauth/token.
Client ID	The OAuth client identifier used for authentication between Mobile Services and the ALM server. Use the service key parameter "clientid".
Client Secret	The OAuth client secret used for authentication between Mobile Services and the ALM server. Use the service key parameter "clientsecret".
ALM Registration Status	The current registration status. The field is initially empty. <ul style="list-style-type: none"> • Pending - the registration request is pending and will be completed within five minutes after the full configuration has been provided. • Registered - the registration was successful and Mobile Services will periodically upload request status with SAP status. • Invalid - the ALM API configuration that was provided is invalid and Mobile Services is not registered with the SAP Cloud ALM server.

3. Select [Save](#).

Once the configuration is in place, an ALM event log is pushed to the ALM service URL. The log can be retrieved and managed from the ALM service URL.

1.5.5.5.8 Application Logs and Trace Files

Set the verbosity for application and component logging, and define how long to keep logs and trace files. You can view all application logs or a subset of your choice, and drill down to view detailed log and trace information if available.

An active user is an individual user of a platform application that connects to the Cloud service at least once during any rolling three-month period. Each active user may only access the Cloud service via a single platform application. An individual accessing the Cloud service via more than one platform application is counted as a separate active user for each platform application.

[Viewing Event Logs \[page 237\]](#)

Use event log information to troubleshoot application problems. Use search criteria to find specific log records and statements.

[Subscribe to Event Log Alerts \[page 240\]](#)

You can subscribe to event logs at the tenant level to receive notifications, and manage the subscriptions through [My Alerts](#).

1.5.5.5.8.1 Viewing Event Logs

Use event log information to troubleshoot application problems. Use search criteria to find specific log records and statements.

Context

Typically, one event log is captured per business operation. However for complex business processes, such as those for Offline and SAP Mobile Cards, data is collected throughout an operation, and written along with the general event log message. The multiple messages are associated using the Correlation ID, and provide you with more detailed information to better isolate a problem.

You can enable detailed logging at the application level as described in *Enabling Event Logs at the App Level*. Note that detailed logging is used for troubleshooting, not for monitoring, so the log is reset after 24 hours. You can manually enable detailed logging again if you need more time for troubleshooting.

You can subscribe to event logs, and manage them via [My Alerts](#). If you create a condition for the subscription via the SAP BTP Alert Notification services (ANS), you can continue to receive alert notifications for three months. You can extend the subscription for another three months using [My Alerts](#).

For some services, you can enable or disable detailed logging either at the application or settings level. In these cases, when you disable detailed logging, error logging is still done. To enable detailed logging for applications, see *Managing Applications*. For the Cloud Build standalone service you can enable detailed event logging and default purge settings via ► [Settings](#) ► [Cloud Build](#) ►. To see the detailed event log related to build process for the cloud build, you need to enable the detailed event log for cloud build feature at the application level (using *Enable Detailed Event Logging*).

Procedure

1. In Mobile Services cockpit, select ► [Analytics](#) ► [Logs](#) ►.
2. To narrow the focus of event logs retrieved from the server, select filters such as [Application ID](#), logging [Level](#), [Correlation ID](#), [Catalog Name](#), [User Name](#), and [Time Frame \(UTC+0000\)](#) (either a predefined time frame, or a custom time frame using the calendar date-picker to define a range). Select [Go](#) to search, and select [Reset Filter](#) to clear your filter entries.
 - To view event log messages for specific applications, select one or more applications from the [Application ID](#) list.
 - To view event log messages for a specific level, select [Level](#), then select one or more logging levels. Leave blank to include all levels.

Logging Levels

Log Level	Description
Debug	For debugging purposes, includes extensive and low-level information.

Log Level	Description
Info	Informational text, used mostly for echoing what has been performed.
Warn	The application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator.
Error	The application can recover from the error, but cannot fulfill the task due to the error.

- To view event log messages that may be related, enter the identifier in [Correlation ID](#). You can enter a full or partial identifier, separated by commas.
- To view event log messages for a specific component, select [Catalog Name](#), then select one or more services. Leave blank to include all services.

System Logging Service Names

Service Name	Description
Mobile App Catalog	Logs system messages that are related to managing the app catalog.
Mobile App Update	Logs messages that are related to application versioning services.
Mobile Application	Logs messages that are related to Native/MDK applications.
Mobile Augmented Reality	Manage client augmented reality resources that can be accessed from mobile applications.
Mobile Client Log Upload	Logs messages that are related to uploading client application log files for analysis on the server.
Mobile Client Resources	Logs messages that are related to resources that can be accessed from mobile applications.
Mobile Client Usage and User Feedback	Logs messages that are related to uploading client usage and user feedback data for analysis on the server.
Mobile Cloud Build	Logs messages that are related to cloud build services.
Mobile Cockpit	Logs system messages that are related to SAP Mobile Services administration.
Mobile Device Security	Logs messages that are related to client device compliance errors.
Mobile Discovery	Logs messages that are related to discovery services.
Mobile Micro App	Logs messages that are related to micro apps.
Mobile Network Trace	Logs messages that are related to network trace.
Mobile Offline Access	Logs system messages that are related to offline access to data on the mobile device.
Mobile Push Notification	Logs system messages that are related to native push actions for iOS and Android devices.
Mobile Sample OData ESPM	Logs system messages that are related to the OData sample service, which can be used during development and testing.
Mobile Settings Exchange	Logs system messages that are related to device registration and the exchange general settings between mobile client and server.

- To view event log messages for requests initiated by a specific user, enter one or more names in [User Name](#). You can enter full or partial names, separated by commas.

- To limit the results within a time frame, select one of the [Time Frame](#) options from the list, or select [Custom Defined](#) and use the calendar date-picker to enter a date range, and click [OK](#).

The search results appear under [Event Logs](#), and are based on your search criteria. The search results count is shown in parentheses.


Column	Description
Time	The time and date stamp of the log entry.
Level	Level value, typically ERROR or WARN or INFO or DEBUG.
Correlation ID	An identifier assigned to each event log. Event logs for complex operations share the same correlation identifier.
Catalog Name	Lists services by name.
User Name	The name of the user associated with the application ID.
Message	The link to detailed log information associated with the execution request.

3. You can apply additional filters to further reduce the search results, such as [Correlation ID](#), [User Name](#), and [Message](#) (you can enter a partial string for the search).

If the filters apply, search results are refreshed and the search results count is reduced.

4. (Optional) To download a text version of the event log file to the Downloads directory, select one or more rows, and click the [Download](#) icon . You can select [Download All](#) to download all of the event logs in the list from the server, or [Download Selected](#) to only download those that you have selected. The [Download](#) icon is grayed-out if no event logs appear in the list. You can save the downloaded file.
5. (Optional) To create a subscription to an event log, select [Subscribe](#). Fill out the settings in [Subscribe Event Log](#), as described in [Subscribe to Event Log Alerts](#).

Once you save the subscription, you can manage it in [My Alerts](#). If you manually create a condition in the SAP Alert Notification service, you can continue to receive alert notifications based on the subscription. See [Managing my Alerts](#) and [Subscribe to Event Log Notices](#) for information.

6. (Optional) To view individual event log messages, select a row, and click the  icon. Under [Event Log Details](#) and [Application and User Details](#) view a summary of event-related information.

Under [Related Events](#) the event message appears. A single event message with a [Seconds Before \(-\) After \(+\)](#) value of 0 indicates that there are no correlated events. Multiple event messages with a range of [Seconds Before \(-\) After \(+\)](#) values indicate that there are several correlated events. Use the values to establish the timeline of what happened leading up to and after the event.

Related Events

Field	Value
Seconds Before (-) After (+)	Indicates when the message occurred: <ul style="list-style-type: none"> • - (minus) – the message occurred before the reported event. • 0 – the event. • + (plus) – the message occurred after the reported event.
Level	The logging level set for the component.

Field	Value
Catalog Name	The component reporting the event.
Message	The message reported.

Related Information

[Enable Detailed Event Logging \[page 280\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

[Managing My Alerts \[page 265\]](#)

[Subscribe to Event Log Notices \[page 262\]](#)

1.5.5.5.8.2 Subscribe to Event Log Alerts

You can subscribe to event logs at the tenant level to receive notifications, and manage the subscriptions through [My Alerts](#).

Context

When you subscribe to event log alerts from Mobile Services cockpit, you can configure the option to send the alerts to the SAP Alert Notification service. This option enables you to control how often and when to send events from SAP Mobile Services to Alert Notification service. You can set a threshold value for when to send events (for example, after one alert, or after five alerts), or set a schedule for when to send events (for example, every hour, every eight hours, or every 24 hours). This reduces the number of repetitive alerts that are sent, so that Alert Notification service is not overwhelmed.

The actual condition for notifications must be created through Alert Notification service. If you create a condition for the subscription via the Alert Notification service, you can continue to receive alert notifications for three months. You can extend the subscription for another three months using [My Alerts](#).

Procedure

1. In Mobile Services cockpit, select [Analytics](#) [Logs](#).
2. You can narrow the focus of event logs to view by selecting filters, such as [Application ID](#), logging [Level](#), [Correlation ID](#), [Feature Name](#), [Time Frame \(UTC+0000\)](#), and [User Name](#). Select [Go](#) to search.
3. To subscribe to the filtered event log, select [Subscribe](#).

In [Subscribe Event Log](#), assign a subscription name and review or edit the subscription. The entries reflect the original filters you selected, but you can modify them.

Event Log Subscription

Setting	Description
Subscription Name	Enter a subscription name that is meaningful to you, such as "com.sap.v2.logout".
Category	Event Log
Application ID	The application identifier.
Level	The log level, such as ERROR.
Feature Name	One or more features that are included in the event log
Send to ANS	<p>Whether to send event log alerts to the SAP Alert Notification service. Select this option if you plan to configure a condition for alert notifications in Alert Notification service.</p> <p>If you enable the Send to ANS option, two additional fields appear. They are used to configure how often and when to send events to Alert Notification service.</p>
Threshold (ANS)	If you enable the Send to ANS checkbox, you can provide a Threshold for when alerts are sent, such as 0, 1 or 5. If the event count number is greater than the threshold number, an alert notification is sent to Alert Notification service.
Schedule (ANS)	If you enable the Send to ANS checkbox, you can select a schedule, such as Every Hour, Every 8 Hours, and Every 24 Hours. Alert notifications are sent to Alert Notification service periodically based on your selection.
Alert Notification Service	If you enable the Send to ANS checkbox, note the Alert Notification service information. You'll need this to manually create the condition in SAP Alert Notification service.

4. Select [Save](#).

Once you save, the subscription appears in [My Alerts](#). The alert subscription remains until you delete it.

5. To receive ongoing alert notifications for the subscription, create a condition in the Alert Notification service. See [Subscribe to Event Log Notices](#).

With the condition, you receive alert notifications for three months. You can extend alert notifications for another three months using [My Alerts](#). See [Managing My Alerts](#). Sample Alert Notification service conditions:

To receive alert notifications for a specific event log subscription:

```
Key: tags.mobile_event_log_alert
Predicate: Is Equal To
Value: MyEventLogSubscription
```

To receive alert notifications for all event log subscriptions in the space/tenant:

```
Key: eventType
Predicate: Is Equal To
```

Value: **MobileServicesEventLogAlert**

Related Information

[Viewing Event Logs \[page 237\]](#)

[Managing My Alerts \[page 265\]](#)

[Subscribe to Event Log Notices \[page 262\]](#)

[Managing My Alerts \[page 265\]](#)

[Subscribe to Event Log Notices \[page 262\]](#)

1.5.5.5.9 Analytic Charts

View analytic charts for the Cloud Foundry environment.

[Viewing Audit Logs \[page 243\]](#)

View audit logs for a record of changes made through Mobile Services cockpit.

[Viewing Server Data Charts \[page 244\]](#)

View analytic charts about data retrieved for server activity.

[Viewing User Data Charts \[page 248\]](#)

View analytic charts composed from data that's retrieved from user's app activity.

[Viewing User Feedback Charts \[page 249\]](#)

(Native apps only) View user feedback reported for a mobile apps.

[Viewing Crash Group Analytics \[page 251\]](#)

(Native Apps Only) View analytics for crash groups and crash dumps that could not be analyzed. If available, you can also download related client log entries and use them to help analyze the problem.

[Subscribe to Crash Alerts \[page 254\]](#)

You can subscribe to crash logs at the tenant level to receive notifications, and manage the subscriptions through *My Alerts*.

[Usage Metering \[page 256\]](#)

View the application usage by license type for the last two or three months. This provides billing history.

1.5.5.5.9.1 Viewing Audit Logs

View audit logs for a record of changes made through Mobile Services cockpit.

Procedure

1. In Mobile Services cockpit, select [Analytics](#) [Auditing](#).
2. To narrow the focus of audit logs retrieved from the server, select filters such as [Application ID](#), [Feature Name](#), [User Name](#), [Change Summary](#), [Time Frame](#) (either a predefined time frame, or a custom time frame using the calendar date-picker to define a range); and select [Go](#). Select [Reset Filter](#) to clear your filter entries.

The search results appear under [Audit Logs](#), and are based on your search criteria. The search results count is shown in parentheses.

Audit Log Columns

Column	Description
Time (UTC+0000)	The time the change took place, in the format YYYY-MM-DD HH:MM:SS:SSS.
User Name	The user who made the change.
Application ID	The application that was changed.
Feature Name	The feature name, such as Mobile Cockpit, Mobile Card Kit, or Mobile Push Notification.
Change Summary	A brief summary of the change made to a service or application, for example, "Service", "Application('com.sap.created.app333')", or "updated existing card type".

3. You can apply additional filters to further reduce the search results, such as [User Name](#) and [Change Summary](#) (you can enter a partial string for the search).

If the filters apply, search results are refreshed and the search results count is reduced.

4. (Optional) To download a text version of the change log file to the Downloads directory, select one or more rows, and click the [Download](#) icon. You can select [Download All](#) to download all of the change logs in the list from the server, or [Download Selected](#) to only download those that you have selected. The [Download](#) icon is grayed-out if no change logs appear in the list. You can save the downloaded file.
5. Select a change record to see its details. Under [Audit Log Details](#), the summary information is provided.

Under [Revision History](#) you can see the attribute that changed, the old value, and the new value.

Revision History Properties

Property	Description
Attribute	The service or application attribute that changed, for example, <code>services/2.name</code> .

Property	Description
Old Value	The last old value. If this is blank, this means a new value was added.
New Value	The new value that was added or made, for example, <code>push</code> . If this is blank, this means the old value was deleted.

1.5.5.5.9.2 Viewing Server Data Charts





View analytic charts about data retrieved for server activity.

Context

Data typically includes counts and averages, and might be expressed as numbers, bar charts, or trend charts. Data categories include:

- Registrations – view analytics that are related to user registrations by app.
- Online Requests – view analytics that are related to online requests.
- Offline Requests – view analytics that are related to offline requests.
- Offline Requests (Android SDK 5.1+, iOS SDK 9.0+) – for applications that were developed with newer Android and iOS SDKs, view additional metrics for offline requests. These metrics rely on features provided in the later SDK versions.
- Push – view analytics that are related to push notifications.

Procedure

1. In Mobile Services cockpit, select  [Analytics](#)  [Server Data](#) .
2. (Optional) You can narrow the focus of server data by setting filters. Select  to set filters, such as [Application ID](#) and [Date](#). Select [Reset](#) to clear your filters.
3. Select [Registrations](#) from the list or tab bar to view analytics that are related to user registrations.
 - Registrations - Total – the total new registrations count for applications.
 - Active App Count – the total active application count with new registrations.
 - Registrations Per App - Total – bar chart that shows the new registrations count by application.
 - Registrations by App – trend line that shows the new registrations count by application. The Y-Axis is the registrations count, and the X-Axis is the date (each application appears in a different color).
 - Active Users Per App - Total – bar chart that shows the number of active users for each application. An "Active User" is one who registers a runtime call for the duration being measured.
 - Active Users by App – trend line that shows the maximum active users count by application. The Y-Axis is the maximum active users count, and the X-Axis is the date (each application appears in a different color).

4. Select [Online Requests](#) from the list or tab bar to view analytics that are related to online requests.
 - Online Requests Count – the total count of online requests.
 - Request Size Average – the average online request payload size in bytes.
 - Response Size Average – the average online response payload size in bytes.
 - Proxy Total Time Average – the average total request time through the proxy service in milliseconds. Any request sent to the proxy service generates this metric.
 - Proxy Backend Time Average – the average request time from the proxy service to the back end in milliseconds. Any request sent to the back end (through the proxy service) generates this metric.
 - Request Size Average – bar chart that shows the average online request payload size in bytes per application.
 - Response Size Average – bar chart that shows the average online response payload size in bytes per application.
 - Proxy Requests (Total vs. Backend) – trend line that shows the online request count for both total and backend. The Y-Axis is the count number, and the X-Axis is the date (total and backend appear in a different color). If a request is sent to the proxy service but not sent to the backend, a metric is generated for 'proxy_total_time' but not for 'proxy_backend_time'. If you see a difference between Total and Backend, this may indicate an issue, for example, something may have happened that prevented the request from being sent to the backend).
 - Online Response Time Average – trend line that shows the average response time for both total and backend in milliseconds. The Y-Axis is the time value, and the X-Axis is the date (total and backend appear in a different color).
 - Online Request Per App Count – trend line that shows the online request count by application. The Y-Axis is the count number, and the X-Axis is the date (the application appears in a different color).
5. Select [Offline Requests](#) from the list or tab bar to view analytics that are related to offline requests.
 - Offline Requests Count – the total offline request count for initial download.
 - Initial Download Time Avg – the average for offline initial download request time in milliseconds.
 - Refresh Time Avg – the average for offline download (refresh) request time in milliseconds.
 - Flush Time Avg – the average for offline upload (flush) request time in milliseconds.
 - Offline Initial Download Time Avg – bar chart that shows the average for offline initial download request time in milliseconds by application.
 - Offline Refresh Time Avg – bar chart that shows the average for offline download (refresh) request time in milliseconds by application.
 - Offline Flush Time Avg – bar chart that shows the average for offline upload (flush) request time in milliseconds by application.
 - Offline Requests Count – trend line that shows the offline request count for initial download, download and upload. The Y-Axis is the count number, and the X-Axis is the date. Initial download, download and upload each appear in a different color.
 - Offline Response Time Avg – trend line that shows the average for offline request time, which includes initial download, download, and upload in milliseconds. The Y-Axis is the value, and the X-Axis is the date. Initial download, download, and upload each appear in different color.
 - Offline Request per App Count – trend line that shows the offline request count by application. The Y-Axis is the count number, and the X-Axis is date (each application appears in a different color).
6. Select [Offline Requests \(Android SDK 5.1+, iOS SDK 9.0+\)](#) from the list or tab bar to view additional analytics that are related to offline requests. This option only appears if the latest SDK versions shown were used to develop the application.

- Requests Count – bar chart that shows the total number of requests by application. The count includes successful and failed requests.
- Initial Download Count – bar chart that shows the total number of initial downloads by application. The count includes successful and failed initial downloads.
- Download Count – bar chart that shows the total number of downloads by application. The count includes successful and failed downloads.
- Upload Count – bar chart that shows the total number of uploads by application. The count includes successful and failed uploads.
- Initial Download Time Average - bar chart that shows the average initial download processing time in milliseconds by application. The count includes the time spent by the client, network and server.
- Download Time Average - bar chart that shows the average download processing time in milliseconds by application. The count includes the time spent by the client, network and server.
- Upload Time Average - bar chart that shows the average upload processing time in milliseconds by application. The count includes the time spent by the client, network and server.
- Synchronize Time Average - bar chart that shows the average processing time spent synchronizing requests in milliseconds for the application. Synchronize refers to the download from the server to client, which follows an upload from client to server. This metric is a composite based on both the download and upload metrics. The count includes the time spent by the client, network and server.
- Requests Count – trend line that shows the request count, which includes initial download, download, and upload. The Y-Axis is the count number, and the X-Axis is the date. Initial download, download, and upload each appear in a different color.
- Request per App Count – trend line that shows the request count for different applications. The Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color).
- Initial Download Time Average – trend line that shows the average initial download processing time in milliseconds. The count includes the time spent by the client, network and server. The Y-Axis is the value, and the X-Axis is the date.
- Download Time Average – trend line that shows the average download processing time in milliseconds. The count includes time spent by the client, network and server. The Y-Axis is the value, and the X-Axis is the date.
- Upload Time Average – trend line that shows the average upload processing time in milliseconds. The count includes time spent by the client, network and server. The Y-Axis is the value, and the X-Axis is the date.
- Synchronize Time Average – trend line that shows the average processing time spent synchronizing requests in milliseconds. Synchronize refers to the download from the server to client, which follows an upload from client to server. This metric is a composite based on both the download and upload metrics. The count includes the time spent by the client, network and server. The Y-Axis is the value, and the X-Axis is the date.

7. Select **Push** from the list or tab bar to view analytics that are related to push notifications.

- iOS Push Count – the total count for iOS pushes that were successful.
- Android Push Count – the total count for Android pushes that were successful.
- Baidu Push Count – the total count for Baidu pushes that were successful.
- Custom Push Count – the total count for pushes to custom push servers. Not available in all countries/regions.
- iOS Push by App Count – trend line that shows the total count of successful iOS pushes by application. The Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color).

- Android Push by App Count – trend line that shows the total count of successful Android pushes by application. The Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color).
- Baidu Push by App Count – trend line that shows the total count of successful Baidu pushes by application. The Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color).
- Custom Push by App Count – trend line that shows the total count of successful pushes to a custom push server by application. The Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color). Not available in all countries/regions.

[Example: Checking Proxy Server Analytics for Destination Name \[page 247\]](#)

This example shows how to check proxy server analytics for destination name.


1.5.5.5.9.2.1 Example: Checking Proxy Server Analytics for Destination Name

This example shows how to check proxy server analytics for destination name.

Context

To do so, configure the proxy destination name as a filter. You can use this approach for similar searches.

Procedure

1. In Mobile Services cockpit, select **Analytics** > **Server Data**.
2. On **Server Data**, select **Online Requests** from the drop-down list, and select the filter button .

Select **Dimensions** and then select **CUSTOM1** from the dimension list.
3. On **Set Filters for <name>**, select the destination name or names to search for, such as **offline**. These represent the destinations that you wish to monitor for CUSTOM1. Select **OK**.

Select the check boxes, such as **offlineDest**, and then select **OK**.
4. **CUSTOM1** appears in the banner as a selected filter.

1.5.5.5.9.3 Viewing User Data Charts

View analytic charts composed from data that's retrieved from user's app activity.

Procedure

1. In Mobile Services cockpit, select [Analytics](#) [User Data](#).
2. (Optional) Use the Add Story Filter/Prompt icon to select filters to narrow the focus of the user data report.

Select [+](#) and [Dimensions](#), and then set filters such as [Application](#), [Application Version](#), [Date](#), [Platform](#), [Platform Version](#), [Device Model](#), [Environment](#), and [Group <x>](#). Select [Reset](#) to clear your filters.

Note

If you select [Group <x>](#), use the [Show User Group Mapping](#) button to view information about the configured user groups. Groups may appear even if data is not collected.

3. Select [Sessions](#) to view charts, which visualize information about usage sessions reported from user devices. A session is defined as the time between a usage `startSession` event and `endSession` event, as coded by the app developer.
 - Sessions per platform – bar chart that shows the session count by platform (such as "iOS", "Mac OS X") and platform version.
 - Sessions per device type – bar chart that shows the session count by device type (for example, "iPhoneSimulator", "iPhone", "MacBookPro"), and device model version.
 - Sessions per app – trend line that shows the session count by application. The Y-Axis is the session count, and the X-Axis is the date (each application appears in a different color).
 - Average session duration per app – trend line that shows the average session duration in seconds by application. The Y-Axis is the average session duration, and the X-Axis is the date (each application appears in a different color).
 - Sessions per platform – trend line that shows the session count by platform and platform version. The Y-Axis is the session count, and the X-Axis is the date. (Each platform/version combination appears in a different color).
 - Average session length – trend line that show the average session duration in seconds by platform. The Y-Axis is the average session duration, and the X-Axis is the date (each platform appears in a different color).
4. Select [Demographics](#) to view charts, which visualize information collected from user devices while sessions run. For some information, such as network information, the app developer decides whether or not to collect it. For other information, such as location data, the user may be able to prevent or allow collection.
 - Total users – the total user count.
 - Bounced sessions – the bounced session count. The bounced session means users don't go past the first page of the app.
 - Sessions on cellular – the session count for those using a cellular network.
 - Sessions on Wi-Fi – the session count for those using a Wi-Fi network.
 - Offline sessions – the session count for those using offline (no network).

- Sessions with location always authorized – the session count for those using a location that is always authorized (developers may implement various standardized categories).
 - Sessions with location not authorized – the session count for those using a location that is not authorized (developers may implement various standardized categories).
 - Sessions with location authorized while in use – the session count for those using a location that is authorized only while the app is in use (developers may implement various standardized categories).
 - Sessions by location (region) (bar/time charts) – bar chart that shows the session count by location. The trend line shows the session count by location. The Y-Axis is the session count, and the X-Axis is date. Each location appears in a different color.
 - Sessions by language (bar/time charts) – the bar chart shows the session count by language. The trend line shows the session count by language. The Y-Axis is the session count, and the X-Axis is date. Each language appears as a different color.
 - Sessions by app versions (bar/time charts) – the bar chart shows the session count by app version. The trend line shows the session count by app version. The Y-Axis is the session count, X-Axis is date. Each app version appears in a different color.
 - Sessions by screen resolution (bar/time charts) – the bar chart shows the session count by screen resolution. The trend line shows the session count by screen resolution. The Y-Axis is the session count, and the X-Axis is date. Each screen resolution appears as a different color.
5. Select [Behavior](#) to view charts that show user's interactions with the app. Data for this option is most meaningful for a single application (that is, combining data for multiple apps doesn't make sense), so use the filter option in step 2 to select an application.
- Active apps – pie chart that shows the started session count by application.
 - User actions – bar chart that shows the user action count by page and action. Examples of user actions include, pressing a button, selecting a row, sliding a bar, opening a window, and so forth.
 - Average page time (seconds) – bar chart that shows the average time spent on each page, expressed in seconds per page.
 - Average page time (percentage) – pie chart that shows the average time spent on each page, expressed as a percentage of the total time per page.
 - Page count – bar chart that shows the page count number by page.
 - Exit page – the last page a user visited before exiting the app.

1.5.5.5.9.4 Viewing User Feedback Charts

(Native apps only) View user feedback reported for a mobile apps.

Prerequisites

The mobile app must be enabled to request, process, and upload user feedback. Supported user feedback includes scale ratings reactions, such as 1-5 stars, and text.

Context

Mobile service cockpit users with developer or manager roles can view feedback from app users, to understand how users responded to user feedback requests.

Procedure

1. In Mobile Services cockpit, select ► [Analytics](#) ► [User Feedback](#) ▾.
2. (Optional) You can narrow the focus of user feedback by setting filters such as application name, application version, platform, platform version, context, device model, user name, date, and more. Select [Reset](#) to clear your filters.
3. Select [Ratings](#) to view the following feedback:
 - Total Ratings – the total count for the feedback received.
 - Average Rating – the average score for all feedback. The value should be between the minimum score and maximum score.
 - Comments Received – the total count for feedback that includes comments.
 - Anonymous Ratings – the total count for feedback from anonymous users.
 - Known User Ratings – the total count for feedback from known users.
 - Daily Ratings – trend line that shows the daily count for feedback. The Left Y-Axis is the count number, and the X-Axis is the date (each application appears in a different color).
 - Average Rating by App – bar chart shows the average score for feedback by application.
 - Average Rating by Context – bar chart shows the average score for feedback by application and context.
 - Average Rating by App – trend line that shows the average score for feedback. The Left Y-Axis is the average score, and the X-Axis is the date (each application appears in a different color).
 - Average Rating by Context – trend line that shows the average score for feedback. The Left Y-Axis is the average score, and the X-Axis is the date (the application and context appear in different colors).
 - Rating by App – bar chart that shows the feedback count number by application and score.
 - Rating by Context – bar chart that shows feedback count number by application, context and score.

1.5.5.5.9.5 Viewing Crash Group Analytics

(Native Apps Only) View analytics for crash groups and crash dumps that could not be analyzed. If available, you can also download related client log entries and use them to help analyze the problem.

Prerequisites

The developer must enable the iOS and Android application to catch exceptions with a global catch-all handler and upload them to Mobile Services server. In Mobile Services cockpit, the application must also be enabled to upload crash logs.

Context

SAP SDK for Android and SAP SDK for iOS may collect the following information for crash logs:

- Platform and version (like “16.6” for iOS)
- Platform architecture (like “Simulator” or “architecture”)
- Stack trace and Thread pool
- Application ID and version
- Build and SDK version
- Device model name and version
- Device locale and screen size
- Record timestamp
- Network connectivity

Crash logs can be an important source of information during the development process. You can enable your test apps to catch crashes and upload the data. Configure Mobile Services to automatically gather crash logs and report failures by group. View the reported data in Mobile Services cockpit, and use to fix issues and ensure quality.

Keep in mind these guidelines:

- You can upload a maximum of 10 crash logs per request to the server. If you exceed this limit, the server reports `Bad Request` (400 error code). The SDK must specify how to handle the limitation and reply to the client.
- The default for both maximum file size and maximum request size is 150 MB, and for the maximum crash log dump size is 1 MB.
- Crash logs are retained for three months and then deleted.
- You can download crash log files, and `dSYM` and mapping configuration files:
 - Download crash log files for the crash reports listed in the Mobile Services cockpit, as described below.
 - Download `dSYM` configuration files from the Mobile Services cockpit as described in *Managing Symbol Files (iOS)*.
 - Download mapping files from the Mobile Services cockpit as described in *Managing Mapping Files (Android)*.

- If client log files related to the crash report are available, you can download the log files to help analyze the problem. A client log is downloaded in ZIP format, and may contain one or multiple log files. The client log feature must be assigned to the mobile app and configured (see *Configuring Mobile Client Log Upload* for information). If no client logs are available, the option to download related client logs does not appear.

You can subscribe to crash logs, and manage them via [My Alerts](#). If you create a condition for the subscription via the SAP BTP Alert Notification service (ANS), you can continue to receive alert notifications for three months. You can extend the subscription for another three months using [My Alerts](#).

Procedure

1. In Mobile Services cockpit, select [Analytics](#) > [Crashes](#).
2. To narrow the focus of crashes to view, select filters such as [Platform](#), [Application ID](#), [Application Version](#), and [Time Frame](#) (either a predefined time frame, or a custom time frame using the calendar date-picker to define a range). The affected applications appear for selection in application ID. If you select a platform and an application ID, the available application versions appear for selection. Time Frame is required. Select [Go](#) to search, and [Reset Filter](#) to clear your filters.

Two charts provide visual information about crashes and users:

- Crashes – the number of crashes that occurred over time.
 - Users – the number of users affected by crashes over time.
3. (Optional) To create a subscription to a crash log, select [Subscribe](#). Fill out the settings in [Subscribe Crash Log](#), as described in *Subscribe to Crash Alerts*.

Once you save the subscription, you can manage it in [My Alerts](#). If you manually create a condition in the SAP Alert Notification service, you can continue to receive alert notifications based on the subscription. See *Managing my Alerts* and *Subscribe to Crash Log Notices* for information.

4. Under [Crash Group](#) view the detected crash groups.

Crash Groups

Column	Description
Title	The title of the crash group.
Count	The crash group count.
Affected Users	The number of affected users.
Platform	The platform, such iOS or Android.
Application Versions	The range of application versions.
Application ID	The application identifier.
Last Report (UTC)	The date and time of the last report update in UTC format (YYYY-MM-DD HH:MM:SS).

Column	Description
Status	<p>The crash dump status by platform. Select a status button to navigate to the specific application for details.</p> <p>For iOS crash dump status:</p> <ul style="list-style-type: none"> • Symbolicated: the crash dump is fully symbolicated. • Processing: the uploaded crash dump is being processed (whether or not dSYMs have been uploaded). • Missing dSYM: the crash dump has been processed. Some debug symbols (dSYMs) are missing, so this crash dump is only partially symbolicated. See <i>Managing Symbol Files (iOS)</i>. Once the required dSYM files have been uploaded, the symbolication process will be retrigged. • Initial: the crash dump has been uploaded and is waiting to be processed. <p>For Android crash dump status:</p> <ul style="list-style-type: none"> • The crash dump is deobfuscated. • The crash dump is obfuscated.

5. Select a crash group to see more details.
6. Select the [Overview](#) tab to view KPIs, charts and the stack trace for the selected crash group.
 - KPIs for:
 - Occurrences – the number of reported occurrences.
 - Users – the number of affected users.
 - Application Version – the application version reporting the crash.
 - Charts for:
 - Affected Device Models – the device models that are reporting the crash.
 - Affected Platform Versions – the operating system platforms that are reporting the crash.
 - Stacktrace – the common stacktrace parts for the crashes, with additional system details.
7. Select the [Reports](#) tab to view information for individual crash dumps by device.
 - a. View a summary of devices reporting crashes

Device Reports

Column	Description
Device	The device type, such as Pixel.
Application Version	The application version associated with the device report.
Platform Version	The operating system platform version, such as <x>.
Date (UTC)	The date and time of the device report in UTC format (YYYY-MM-DD HH:MM:SS).

- b. Select one of the device reports to view its details.

- Report Details – device details such as date, device model, platform version, application ID and version, screen size, language, SDK product (such as iOS or Android), and SDK version.
If available, select [↓](#) to download related client log entries for one client log ID. If there is no data, the download option is not available.
- Raw Data – crash data for the device. Select [↓](#) to download data.

Related Information

[Managing Symbol Files \(iOS\) \[page 86\]](#)

[Managing My Alerts \[page 265\]](#)

[Subscribe to Crash Alerts \[page 254\]](#)

[Subscribe to Crash Log Notices \[page 261\]](#)

[Managing Mapping Files \(Android\) \[page 87\]](#)

[Configuring Mobile Client Log Upload \[page 58\]](#)

1.5.5.5.9.6 Subscribe to Crash Alerts

You can subscribe to crash logs at the tenant level to receive notifications, and manage the subscriptions through [My Alerts](#).

Context

When you subscribe to crash alerts from Mobile Services cockpit, you can configure the option to send the alerts to the SAP Alert Notification service. This enables you to control how often and when to send events from SAP Mobile Services to Alert Notification service. You can set a threshold value for when to send events (for example, after one alert, or after five alerts), or set a schedule for when to send events (for example, every hour, every eight hours, or every 24 hours). This reduces the number of repetitive alerts that are sent, so that Alert Notification service is not overwhelmed.

The actual condition for notifications must be created through SAP Alert Notification service. If you create a condition for the subscription via the Alert Notification service, you can continue to receive alert notifications for three months. You can extend the subscription for another three months using [My Alerts](#).

Procedure

1. In Mobile Services cockpit, select [▶ Analytics ▶ Crashes ▶](#).
2. You can narrow the focus of crashes by selecting the [Platform](#), [Application ID](#), [Application Version](#), and [Time Frame](#) filters. Select [Go](#) to search.

3. To subscribe to the filtered crash log, select [Subscribe](#).

In [Subscribe Crash Log](#), assign a subscription name and review or edit the subscription. The entries reflect the original filters you selected, but you can modify them.

Crash Log Subscription

Setting	Description
Subscription Name	Enter a subscription name that is meaningful to you, such as "TEST.sample.basic".
Category	Crash Log
Application ID	The application identifier, such as TEST.sample.basic.
Platform	The platform, such as Android or iOS.
Send to ANS	<p>Whether to send crash log alerts to the SAP Alert Notification service. Select this option if you plan to configure a condition for alert notifications in SAP Alert Notification service.</p> <p>If you enable the Send to ANS option, two additional fields appear. They are used to configure how often and when to send events to Alert Notification service.</p>
Threshold (ANS)	If you enable the Send to ANS checkbox, you can provide a Threshold for when alerts are sent, such as 0, 1 or 5. If the event count number is greater than the threshold number, an alert notification is sent to Alert Notification service.
Schedule (ANS)	If you enable the Send to ANS checkbox, you can select a schedule, such as Every Hour, Every 8 Hours, and Every 24 Hours. Alert notifications are sent to Alert Notification service periodically based on your selection.
Alert Notification Service	If you enable the Send to ANS checkbox, note the Alert Notification service information. You'll need this to manually create the condition in SAP Alert Notification service.

4. Select [Save](#).

Once you save, the subscription appears in [My Alerts](#). The alert subscription remains until you delete it.

5. To receive ongoing alert notifications for the subscription, create a condition in the SAP Alert Notification service. See [Subscribe to Crash Log Notices](#).

With the condition, you receive alert notifications for three months. You can extend alert notifications for another three months using [My Alerts](#). See [Managing My Alerts](#). Sample Alert Notification service conditions:

To receive alert notifications for a specific crash log subscription:

```
Key: tags.mobile_crash_log_alert
Predicate: Is Equal To
Value: MyCrashLogSubscription
```

To receive alert notifications for all crash log subscriptions in the space/tenant:

```
Key: eventType  
Predicate: Is Equal To  
Value: MobileServicesCrashLogAlert
```

Related Information

[Managing My Alerts \[page 265\]](#)

[Subscribe to Crash Log Notices \[page 261\]](#)

[Managing My Alerts \[page 265\]](#)

[Subscribe to Crash Log Notices \[page 261\]](#)

1.5.5.5.9.7 Usage Metering

View the application usage by license type for the last two or three months. This provides billing history.

Prerequisites

This feature is available for "standard (Users)" and "resources" and "build-code" license types, and reporting is done at the space level.

Note

If you plan to use SAP Build Code, see [What is SAP Build Code](#) to learn more about the service.

If you are using this service as part of SAP Build Code, follow the [SAP Build Code Initial Setup](#) instructions.

Context

From Mobile Services cockpit, you can view the usage history that is used to calculate billing. The cumulative user and resource count is provided by the month for the last three months; the build code count is provided by month, for the last two months. You can also download raw user and resource metering data for a specific date/time range in .csv format, to see application usage. This feature provides billing transparency, and usage information. Note that there is a one-day lag for the last day of the month; its data is not included until the following day. For example, to see data for September 30th, click the September bar on October 2.

Procedure

1. In Mobile Services cockpit, select **Analytics** > **Usage Metering**.
2. For **License Type**, select one of the following:
 - **Standard** to view usage history for the "standard (Users)" license plan.
 - **Resources** to view usage history for the "resources" license plan.
 - **Build-code** to view usage history for the "build-code" license plan.
3. Depending on your license type selection, a graphic shows the user or resource or build-code count for the past three months.
4. Click on a time bar to see details for the month. In the **Monthly <name>** pane, the table is updated to show the registered users or resources counted within the month.
5. (Optional) In the **Monthly <name>** pane, you can search for a specific user or resource. You can enter a partial string.
6. Select **Download Raw Data** in the upper right corner to download raw data locally in a .csv file. Raw data includes the applications that the selected users or resources have accessed.
 - a. In **Custom Download Data Range (UTC+0000)** enter a date range within the last two or three months, and click **OK**.

The file is downloaded to your local environment, either:

- users_metering_history_<date_range>
- resources_metering_history_<date_range>
- build-code_metering_history_<date_range>

- b. View the metering history for the date range selected.

Metering History

Column	Description
Report Date	The date that the user or resource accessed the application.
Application ID	The identifier for the application that was accessed by the registered user or resource.
User Name	The user name for the registered user that accessed the application.
Resource Name	The resource name for the individual or asset that accessed an application. The resources option enables customers to view user login details for Mobile Services. The build-code option enables customers who purchased Capacity Units for the Build Code Plan, to view user login details for Mobile Services in a similar way.

- c. You can review the data to learn more about the applications that a user or resource has accessed. Optionally you can save the file.

1.5.5.5.10 Managing Alerts

You can configure alert notifications for SAP BTP resources and Mobile Services using the SAP Alert Notification service. You can also subscribe to event and crash logs from Mobile Services cockpit and configure alert notifications in the Alert Notification service.

To use the SAP Alert Notification service, you must enable the service and configure Subscriptions and Alerts. For example, you could create conditions to receive notifications for one or more applications, or for release upgrades. The notifications can be consumed using email, Slack, and other channels.

From Mobile Services cockpit, you can subscribe to event and crash logs, and create a condition in the Alert Notification service to receive alert notifications. Use [My Alerts](#) to manage event and crash logs subscriptions, and view release notes.

[Subscribe to Mobile Service Alerts \[page 258\]](#)

Use the SAP Alert Notification service for SAP BTP to receive notifications about Mobile Services alerts.

[Subscribe to Release Upgrade Notices \[page 259\]](#)

Use the SAP Alert Notification service for SAP BTP to receive notifications when a new version of Mobile Services is available.

[Subscribe to Crash Log Notices \[page 261\]](#)

Use the SAP Alert Notification service (ANS) to receive ongoing Mobile Services crash log notifications.

[Subscribe to Event Log Notices \[page 262\]](#)

Use the SAP Alert Notification service (ANS) to receive ongoing Mobile Services event log notifications.

[Subscribe to MDK Project Build Notices \[page 263\]](#)

Use the SAP Alert Notification service (ANS) to receive result notifications when building AppUpdate Mobile Development Kit projects for Mobile Services.

[Subscribe to Certificate Expiration Notices \[page 264\]](#)

Use the SAP Alert Notification service (ANS) to receive ongoing notifications when certificates are to expire.

[Managing My Alerts \[page 265\]](#)

Use [My Alerts](#) to manage Mobile Services subscriptions to event or crash log events at the tenant level, and to access release notes. You can create conditions in SAP Alert Notification service to receive ongoing notifications for these alerts.

1.5.5.5.10.1 Subscribe to Mobile Service Alerts

Use the SAP Alert Notification service for SAP BTP to receive notifications about Mobile Services alerts.

Prerequisites

Users must enable the SAP Alert Notification service, and configure Subscriptions and Alerts (subscriptions and alerts cannot be managed through Mobile Services cockpit).

Alerts must be enabled, and alert levels must be configured for the application.

Context

The SAP Alert Notification service enables users to receive and create custom alerts and notifications for SAP BTP resources, and consume them using the system of their choice such as email, Slack, and other channels. See [SAP Alert Notification service for SAP BTP](#) for information about the service.

Procedure

1. (Users) To receive notifications, create a condition in the SAP Alert Notification service in the same space:

```
Key: tags.mobile_alert_app  
Predicate: Is Equal To  
Value: appId
```

2. (Users) To receive alerts for all applications, create a condition:

```
Key: eventType  
Predicate: Is Equal To  
Value: MobileServicesApplicationAlert
```

1.5.5.5.10.2 Subscribe to Release Upgrade Notices

Use the SAP Alert Notification service for SAP BTP to receive notifications when a new version of Mobile Services is available.

Prerequisites

Users must enable the SAP Alert Notification service, and configure Subscriptions and Alerts (subscriptions and alerts cannot be managed through Mobile Services cockpit).

Context

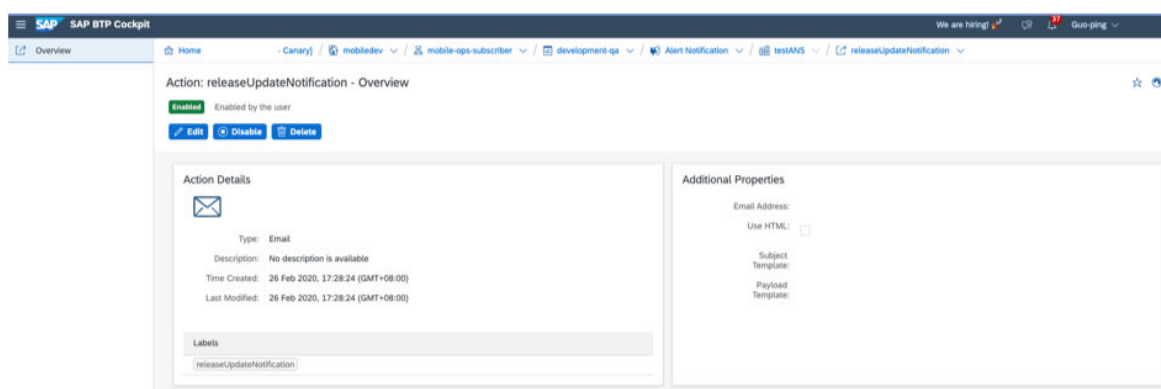
The Alert Notification service enables users to receive and create custom alerts and notifications for SAP BTP resources, and consume them using the system of their choice such as email, Slack, and other channels. See [SAP Alert Notification service for SAP BTP](#) for information about the service.

(Users) When setting up the Subscription, you'll need the following condition configuration information:

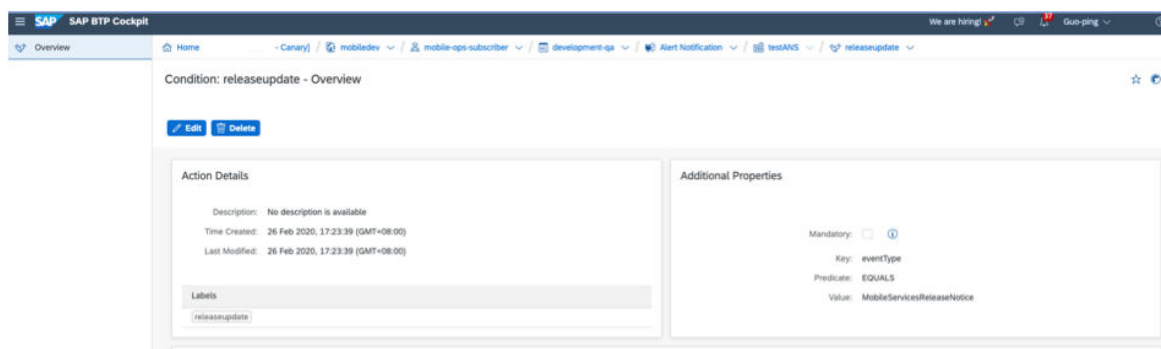
```
Key: eventType  
Predicate: Is Equal To  
Value: MobileServicesReleaseNotice
```

Procedure

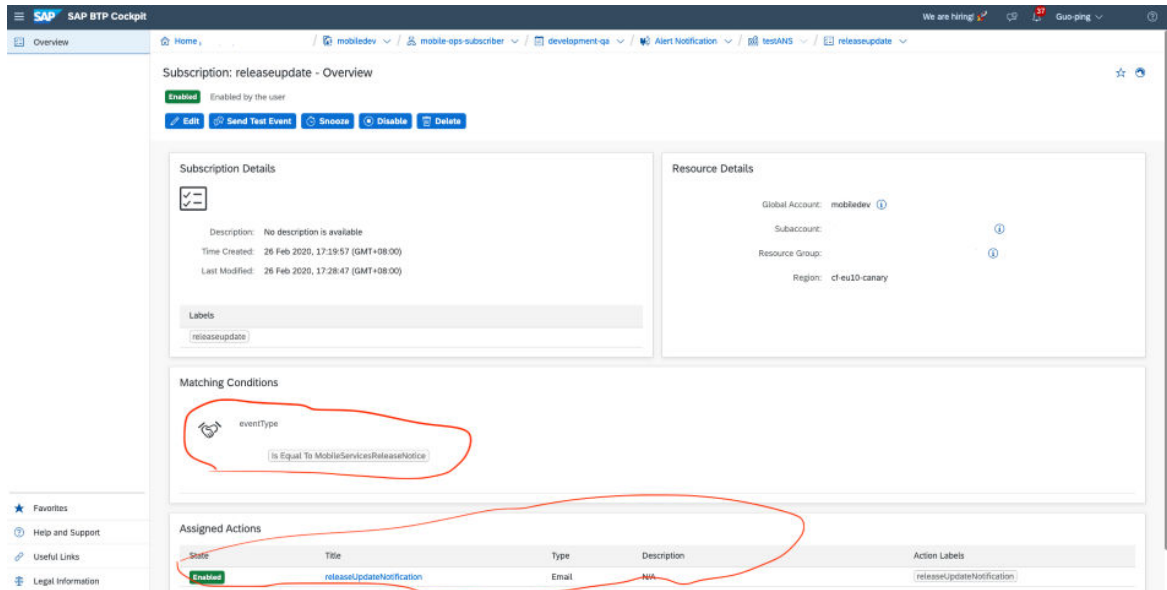
1. From SAP BTP cockpit, create an Alert Notification Service instance.
2. From the instance, create an action to send email.



3. From the instance, create a condition.



4. Create a subscription to link the action and the condition together.



To learn more: [Event Subscriptions](#).

- Once configured, when the Mobile Services release is updated, a release update notification is sent to the email address. At any time you can disable the subscription.

1.5.5.5.10.3 Subscribe to Crash Log Notices

Use the SAP Alert Notification service (ANS) to receive ongoing Mobile Services crash log notifications.

Prerequisites

You must subscribe to a crash log in Mobile Services cockpit as described in *Subscribe to Crash Alerts*. When you create the subscription and enable the ANS checkbox, you receive Alert Notification information to help you create a condition in the Alert Notification service.

Context

Create a condition for the subscription via the SAP BTP Alert Notification service, and you can continue to receive alert notifications for three months. See [SAP Alert Notification service for SAP BTP](#) for information about the service.

You can extend the subscription for another three months using [My Alerts](#). See *Managing My Alerts*.

Procedure

1. (Users) To receive notifications on crash log type subscription, create a condition in the SAP Alert Notification service in the same space::

```
Key: tags.mobile_crash_log_alert  
Predicate: Is Equal To  
Value: subscription_name
```

2. (Users) To receive notifications for all crash log subscriptions, create a condition in the SAP Alert Notification service in the same space:

```
Key: eventType  
Predicate: Is Equal To  
Value: MobileServicesCrashLogAlert
```

Related Information

[Subscribe to Crash Alerts \[page 254\]](#)

[Subscribe to Crash Alerts \[page 254\]](#)

[Managing My Alerts \[page 265\]](#)

1.5.5.5.10.4 Subscribe to Event Log Notices

Use the SAP Alert Notification service (ANS) to receive ongoing Mobile Services event log notifications.

Prerequisites

You must subscribe to an event log in Mobile Services cockpit as described in *Subscribe to Event Log Alerts*. When you create the subscription and enable the ANS checkbox, you receive Alert Notification information to help you create a condition in the Alert Notification service.

Context

Create a condition for the subscription via the SAP BTP Alert Notification services, and you can continue to receive alert notifications for three months. See [SAP Alert Notification service for SAP BTP](#) for information about the service.

You can extend the subscription for another three months using [My Alerts](#). See *Managing My Alerts*.

Procedure

1. (Users) To receive notifications for event log type subscriptions, create a condition in the SAP Alert Notification service in the same space:

```
Key: tags.mobile_event_log_alert  
Predicate: Is Equal To  
Value: subscription_name
```

2. (Users) To receive notifications for all event log subscriptions, create a condition in the SAP Alert Notification service in the same space:

```
Key: eventType  
Predicate: Is Equal To  
Value: MobileServicesEventLogAlert
```

Related Information

[Viewing Event Logs \[page 237\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

[Managing My Alerts \[page 265\]](#)

1.5.5.5.10.5 Subscribe to MDK Project Build Notices

Use the SAP Alert Notification service (ANS) to receive result notifications when building AppUpdate Mobile Development Kit projects for Mobile Services.

Prerequisites

Users must enable the SAP Alert Notification service, as well as configure Subscriptions and Alerts (subscriptions and alerts cannot be managed through SAP mobile service cockpit).

Context

The SAP Alert Notification service enables users to receive and create custom alerts and notifications for SAP Business Technology Platform resources, and consume them using the system of their choice, such as email, Slack, and other channels. See [SAP Alert Notification service for SAP BTP](#) for information about the service. For more information see [Managing Subscriptions](#).

Procedure

1. (Users) When setting up the Subscription, you'll need the following conditions configuration information:

```
Key: subject Predicate: EQUALS Value: SAP Mobile Services AppUpdate MDK  
Project Build Result
```

```
Key: body Predicate: CONTAINS Value: Subaccount Id: <subaccountId>
```

2. AppUpdate MDK project build results are sent for both success or failure. The difference is that successful results report [INFO] and [NOTIFICATION] types in the subject, and failure results report [WARNING] and [ALERT] types.

Use the notifications to view and track issues.

1.5.5.5.10.6 Subscribe to Certificate Expiration Notices

Use the SAP Alert Notification service (ANS) to receive ongoing notifications when certificates are to expire.

Context

You can define the expiration time for certificates in Mobile Service, for example: `service-instance`, `service-proxy`, and `service-push`. To help administer certificate expiration, you can also add certificate expiration alerts in ANS so that you can be alerted if the expiration date is coming.

Create a condition for the subscription via the SAP BTP Alert Notification service, and you can continue to receive alert notifications for three months. See [SAP Alert Notification service for SAP BTP](#) for information about the service.

You can extend the subscription for another three months using [My Alerts](#). See *Managing My Alerts*.

Procedure

1. To receive notifications for a `service-instance` subscription, create a condition in the SAP Alert Notification service in the same space.

CertExpirationAlertProperties for `service-instance`:

```
resourceType = "SAP Mobile Services"  
eventType = "MobileServicesCertificateExpiration"  
alertAppTag = "ms_app"  
alertSubject = "Certificate will expire soon"  
serviceTag = "ms_service"  
expiredDuration = 30
```

For service instance, certificate expiration alert can be used on service keys with an X509 certificate.

2. To receive notifications for a `service-proxy` subscription, create a condition in the SAP Alert Notification service in the same space.

CertExpirationAlertProperties for `service-proxy`:

```
resourceType = "SAP Mobile Services"
eventType = "MobileServicesCertificateExpiration"
alertAppTag = "ms_app"
alertSubject = "Certificate will expire soon"
serviceTag = "ms_service"
expiredDuration = 30
```

For `service proxy`, certificate expiration alert can be used on the trust store and key store of endpoint.

3. To receive notifications for a `service-push` subscription, create a condition in the SAP Alert Notification service in the same space.

CertExpirationAlertProperties for `service-push`:

```
resourceType = "SAP Mobile Services"
eventType = "MobileServicesCertificateExpiration"
alertAppTag = "ms_app"
alertSubject = "Certificate will expire soon"
serviceTag = "ms_service"
expiredDuration = 30
```

For `service push`, certificate expiration alert can only be used on APNS. The global push configuration should not be configured.

1.5.5.5.10.7 Managing My Alerts

Use [My Alerts](#) to manage Mobile Services subscriptions to event or crash log events at the tenant level, and to access release notes. You can create conditions in SAP Alert Notification service to receive ongoing notifications for these alerts.

Prerequisites

You must enable the SAP Alert Notification service and configure Subscriptions and Alerts to receive ongoing alert notifications for event and crash logs. For convenience, the format for the alert condition is provided when you subscribe to an event or crash logs.

Procedure

1. Mobile Services cockpit, select [My Alerts](#).
2. You can narrow the focus of events to view by setting filters, such as [Time Frame](#) and [Category](#) (server side search); and [Send to ANS](#) and subscription name (local search). You can also select [Filter out 0 Message Subscriptions](#) to filter out message subscriptions without message titles, to clean up the view (local search). Select [Go](#) to start the search, and [Reset Filter](#) to clear your filter entries.

A list of events appears based on the filters selected. Any row with a blue highlight to the left indicates that new or unread message were received since the subscription was last viewed.

My Events

Column	Description
Latest Message Time (UTC)	The time of the last message received for the subscription.
Count	The total count of messages received for the subscription in the time frame selected.
Category	The event category for the subscription, including Event Log, Crash Log, Release Note, or All.
Latest Message Title	The latest message title received for the subscription.
Subscription Name	The subscription name assigned to an event.
Send to ANS	Whether to send the event to the SAP Alert Notification service.
ANS Expires (UTC)	Identifies when the Alert Notification service subscription expires.
Actions	Actions you can take such as edit, delete, or extend the subscription. If a subscription is a month from expiration, you can extend it another three months. The new date is reflected in the ANS Expires column.

3. Select a row to see details.

- Select an [Event Log](#) row to open the log in ► [Analytics](#) ► [Logs](#) ▾, with your filters set. You can subscribe to the event, or view details. See [Subscribe to Event Log Alerts](#) to learn more.
- Select a [Crash Log](#) row to open the log in ► [Analytics](#) ► [Crashes](#) ▾, with your filters set. You can subscribe to the event, or view details. See [Subscribe to Crash Alerts](#) to learn more.
- Select a [Release Note](#) row to open the [Release Note](#) page for a release note version. The [Release Note Overview](#) provides links to information.

Related Information

[Viewing Event Logs \[page 237\]](#)

[Subscribe to Crash Alerts \[page 254\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

[Subscribe to Crash Alerts \[page 254\]](#)

[Subscribe to Event Log Alerts \[page 240\]](#)

[Subscribe to Event Log Notices \[page 262\]](#)

[Subscribe to Crash Log Notices \[page 261\]](#)

1.5.5.5.11 Managing Application Themes

(Native/MDK only) As an administrator you can manage application themes from Mobile Services cockpit and indicate a default theme for mobile apps.

Prerequisites

Themes must be created using the SAP Theme Designer and the SAP Horizon Dark / Light theme as the base. Only changes to the background color and text color can be made. The theme must be exported to ZIP format using the export option *Optional Settings (for Experts) > Source files + CSS*, and deselecting the UI technologies.

The `theme.zip` file contains both light and dark versions of the theme, including company logos in JPG format, a JSON source file for Android apps, and an NSS source file for iOS apps.

The Mobile Services server validates the `theme.zip` file upon upload to the storage service, to ensure security and artifact validity. These validation rules are checked:

- The uploaded theme file must be in a .ZIP format.
- The uploaded theme file maximum size is 10M.
- The uploaded theme designer ZIP must be of the `sap_horizon` type.
- The virus scan must pass for the uploaded theme ZIP file.
- The user can upload up to two theme ZIP files in one request.
- The following ZIP file content checks must pass:
 - The `Base/baseLib/<theme_name>/ .theming` file must be included and the file content must be a valid JSON format.
The JSON content must contain the following properties and corresponding values:
 - `sId`
 - `sLabel`
 - `oExtends`
 - `aBadgeParameters`
 - The `Base/baseLib/<theme_name>/ .less_variables.less` file must be included.
- If any validation fails, an event log is created for it. Check the Event Log using the Mobile Settings Exchange filter for the storage service, as described in *Viewing Event Logs*.

Context

Themes change the appearance of the mobile client on the user's device. The Mobile Services server uses the settings to apply themes to the mobile app client. Each theme definition includes a light and dark version of the theme. Mobile client users can set the active theme from the device. Themes are managed at the space level.



Themes must be defined using the SAP Theme Designer's SAP Horizon Dark / Light theme, and distributed in ZIP format. Once uploaded, you can set the active theme. If you decide to deselect themes, the SDK application does not get the theme from the Mobile Services server.

Note that for a single application cockpit (Software as a Service, SaaS), only one customized theme is available. The theme is automatically enabled for the application, once the customized theme is enabled. You can update and delete themes. If the theme is deleted, the SDK application uses the initial default theme instead of getting the theme from the Mobile Services server. The appearance information is cleared, and you can import another theme.

After the initial upload to [App Theme Manager](#), to update themes you can either:

- Use the SAP Theme Designer to modify the theme, export it to a new ZIP file, and upload the file in the [App Theme Manager](#).
- Download the ZIP file that contains the NSS and JSON files from the [App Theme Manager](#) (provider) or from [Appearance](#) (SaaS application). Modify the final processed theme ZIP file directly, and upload it when finished.

Procedure


1. In Mobile Services cockpit, select  [Settings](#) > [App Theme Manager](#) .
2. View the list of themes that have been uploaded.



Application Theme Settings

Column	Description
Name	The unique name associated with a theme listed in the cockpit theme list.
ID	The identifier that was assigned in the SAP Theme Designer. Note that an ID may be reused, for example, if a Theme Designer ZIP file contains both light and dark themes.
Title	The descriptive theme title, such as Custom High Contrast Morning, or Custom Evening.
Color	The light and dark color palettes defined for the theme.
Creation Date (UTC+0000)	The date and time the theme was created, in the format YYYY-MM-DD HH:MM:SS:SS.
User Selection	Whether the theme appears in the application Appearance theme list. If the theme appears in the theme list, the user can select it to make the theme active for the application.
As Default	Whether the theme should be used as the default theme. A default theme becomes the active theme for all the applications in the space.
Actions	The actions the administrator can take, such as download the theme ZIP file locally, update the theme with the latest version from SAP Theme Designer, or delete the theme.

3. Select  to add a new theme. In [Add a Theme](#), provide the theme definition and select [Save](#).

Theme Definition

Property	Description
Theme Name	Provide a descriptive name to identify the app theme. Both light and dark versions of the theme are identified with this name. You cannot change the theme name once you save.
Light Theme	Browse to select the lighter version of an app theme.
Dark Theme	Browse to select the darker version of an app theme. To learn more about implementing Android dark theme, see Dark Theme  .

4. Select  to download a theme locally. The `theme.zip` file is downloaded to your local device. Optionally, you can update the files to change the themes.
5. Select [Update](#) to update the theme using the latest version. You can:
 - [Upload Theme Designer zip](#) - select this option to upload the latest light or dark theme version from SAP Theme Designer.
 - [Upload Processed Theme zip](#) - select this option to upload a ZIP file that you have modified locally to change the theme definition.
6. Select  to delete the theme.




Related Information

[Managing Themes at the App Level \[page 269\]](#)

1.5.5.5.11.1 Managing Themes at the App Level

(Native/MDK) You can manage the themes available for the selected application.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. Select [Client Configuration](#).
4. Under [Appearance](#), make sure [Enable Custom Theme](#) is selected.
5. View the list of themes that have been uploaded. The theme list reflects settings made in the [App Theme Manager](#). See [Managing Application Themes](#).
6. In the [Active](#) column, select the theme to be active for the selected application, and then [Save](#). Both the light and dark versions of the theme will be available for the mobile client user to select.

Related Information

[Managing Application Themes \[page 267\]](#)

1.5.5.5.11.2 Application Theme Color Mapping

Describes the color mapping that is used by Application Theme Manager, between web colors and runtime applied colors for iOS and Android.

iOS Color Mapping

iOS Color Mapping

Web Color	Light Theme	Dark Theme
sapBrandColor	@tintColor	@tintColor_darkBackground
	@tintColor_elevatedLightBackground	@tintColor_elevatedDarkBackground
	@tintColor2	@tintColor2_darkBackground
	@tintColor2_elevatedLightBackground	@tintColor2_elevatedDarkBackground
	@tintColorTapState	@tintColorTapState_darkBackground
	@tintColorTapState_elevatedLightBackground	@tintColorTapState_elevatedDarkBackground
sapShellColor	@header	@header_darkBackground
	@header_elevatedLightBackground	@header_elevatedDarkBackground
	@headerBlended	@headerBlended_darkBackground
	@headerBlended_elevatedLightBackground	@headerBlended_elevatedDarkBackground
	@barTransparent	@barTransparent_darkBackground
	@barTransparent_elevatedLightBackground	@barTransparent_elevatedDarkBackground
sapBaseColor	@primaryBackground	@primaryBackground_darkBackground
	@primaryBackground_elevatedLightBackground	@primaryBackground_elevatedDarkBackground

Web Color	Light Theme	Dark Theme
	@tertiaryBackground	@tertiaryBackground_darkBackground
	@tertiaryBackground_elevatedLightBackground	@tertiaryBackground_elevatedDarkBackground
	@secondaryGroupedBackground	@secondaryGroupedBackground_darkBackground
	@secondaryGroupedBackground_elevatedLightBackground	@secondaryGroupedBackground_elevatedDarkBackground
sapBackgroundColor	@secondaryBackground	@secondaryBackground_darkBackground
	@secondaryBackground_elevatedLightBackground	@secondaryBackground_elevatedDarkBackground
	@primaryGroupedBackground	@primaryGroupedBackground_darkBackground
	@primaryGroupedBackground_elevatedLightBackground	@primaryGroupedBackground_elevatedDarkBackground
	@tertiaryGroupedBackground	@tertiaryGroupedBackground_darkBackground
	@tertiaryGroupedBackground_elevatedLightBackground	@tertiaryGroupedBackground_elevatedDarkBackground
sapShell_TextColor	@customColor1	@customColor1_darkBackground
	@customColor1_elevatedLightBackground	@customColor1_elevatedDarkBackground
	@navBarTitleLabel	@navBarTitleLabel_darkBackground
	@navBarTitleLabel_elevatedLightBackground	@navBarTitleLabel_elevatedDarkBackground
sapTextColor	@primaryLabel	@primaryLabel_darkBackground
	@primaryLabel_elevatedLightBackground	@primaryLabel_elevatedDarkBackground
sapContent_LabelColor	@secondaryLabel	@secondaryLabel_darkBackground
	@secondaryLabel_elevatedLightBackground	@secondaryLabel_elevatedDarkBackground
	@tertiaryLabel	@tertiaryLabel_darkBackground

Web Color	Light Theme	Dark Theme
	@tertiaryLabel_elevatedLightBack-ground	@tertiaryLabel_elevatedDarkBack-ground
	@quarternaryLabel	@quarternaryLabel_darkBackground
	@quarternaryLabel_elevatedLightBack-ground	@quarternaryLabel_elevatedDarkBack-ground
sapNegativeColor	@negativeLabel	@negativeLabel_darkBackground
	@negativeLabel_elevatedLightBack-ground	@negativeLabel_elevatedDarkBack-ground
sapCriticalColor	@criticalLabel	@criticalLabel_darkBackground
	@criticalLabel_elevatedLightBack-ground	@criticalLabel_elevatedDarkBack-ground
sapPositiveColor	@positiveLabel	@positiveLabel_darkBackground
	@positiveLabel_elevatedLightBack-ground	@positiveLabel_elevatedDarkBack-ground

Android Color Mapping

Android Color Mapping

Web Color	Light Theme	Dark Theme
sapBrandColor	s2	s2_dark
	t4	t4_dark
	t6	t6_dark
	t7	t7_dark
	b3	b3_dark
	b4	b4_dark
	r3	r3_dark
sapShellColor	s6	s6_dark
sapBaseColor	s1	s1_dark
sapBackgroundColor	s0	s0_dark

Web Color	Light Theme	Dark Theme
	s4	s4_dark
sapShell_TextColor	t8	t8_dark
sapTextColor	t1	t1_dark
sapContent_LabelColor	t2	t2_dark
	t3	t3_dark
sapNegativeColor	semantic_negative	semantic_negative_dark
sapCriticalColor	semantic_critical	semantic_critical_dark
sapPositiveColor	semantic_positive	semantic_positive_dark

1.5.5.5.12 Managing App Catalog Settings

As an administrator, you can manage app catalog settings.

The app catalog enables developers to manage internal applications during the development and test cycle. As a developer administrator you can access the App Lab service to manage versions; create enterprise mobility management (EMM) destinations that can be used by devices that aren't registered with Mobile Services; and manage service keys that enable programmatic access to application versions that have been published.

[Manage App Lab \[page 274\]](#)

Make application versions available for testing.

[Manage EMM Destinations \[page 275\]](#)

Administrators can create enterprise mobility management (EMM) destinations that can be used by devices that aren't registered with the system.

[Manage Service Keys \[page 276\]](#)

Define service keys to enable a third party to access application versions via the API.

Related Information

[Managing Application Versions Using App Lab \[page 175\]](#)

1.5.5.5.12.1 Manage App Lab

Make application versions available for testing.

Prerequisites

Update an application version as described in *Managing Application Versions Using App Lab*.

Context

Developers and administrators can publish an app version to the App Lab service, which deploys the app version into your Cloud Foundry space. Once the app's been deployed, you can notify users that it's available for testing.

Procedure

1. In Mobile Services cockpit, select [Settings](#) > [App Catalog](#) > [App Lab](#) >

Note

Remember, if you're a SAP BTP Cloud Foundry space member administrator, your space member roles also apply when you access the Mobile Services cockpit [Settings](#) > [App Catalog](#) > [App Lab](#) > option.

As an administrator, you can indicate whether users without the "Viewer" scope have read access to App Lab. By default, [Restrict App Lab to authorized users](#) is selected, which prevents users without the "Viewer" scope from logging in to App Lab. If you unselect the option, users who aren't assigned the "Viewer" scope can log in with read access.

2. In [App Lab](#), click [Initialize](#). You need to do this only once for the Cloud Foundry space you are administering. It may take a moment for the app to become available at the address shown on the page; you may see a 404 Not Found message during this time.
3. Copy the link that appears on the page, and send it to your testers.
4. An [Upgrade](#) button in the upper-right corner indicates that App Lab needs to be updated. You may also see the message *Upgrade is required for App Lab*. Click [Upgrade](#).

Select [Upgrade](#) to proceed. The upgrade takes about five minutes.

1.5.5.5.12.2 Manage EMM Destinations

Administrators can create enterprise mobility management (EMM) destinations that can be used by devices that aren't registered with the system.

Context

Procedure

1. In Mobile Services cockpit, select **Settings** > **App Catalog** > **EMM**.
2. Under **Destinations**, you can view current destinations that are registered with enterprise mobility management. Use **Search** to search for a subset of destinations.
 - Name
 - Description
 - Vendor
 - Actions
3. Select **+** to add a destination.
 1. In **Add Destination**, configure the destination.

Destination Properties

Property	Description
Select EMM Vendor	An enterprise mobility management vendor, such as "VMware."
Name	The destination name.
Description	(Optional) The destination description.
EMM Host URL	A URL associated with the host, such as https://cn<xxx>.airwatchportals.com.
Organization Group	The organization group associated with this destination.
Group ID	(Optional) A unique group ID. Enables you to distinguish between groups with the same name.
API Key	The API key that is defined by the remote system, which enables devices to interact as a client of the remote API.
User Name	The user name of the account used to log in to the host URL.

Property	Description
Password	The password of the account used to log in to the host URL.

2. Select [Test Connection](#) to test the destination configuration.
3. Select [Save](#).

1.5.5.5.12.3 Manage Service Keys

Define service keys to enable a third party to access application versions via the API.

Prerequisites

Make a list of apps that are available via OData that can be downloaded. Make sure the list and downloaded OData APIs are authorized by an API key from the remote system. The App Lab must be initialized.

Context

Service keys grant external users programmatic access to list and download published application versions. This allows integration with third-party tools such as enterprise mobility management (EMM) suites and test automation.

Procedure

1. In Mobile Services cockpit, select [Settings](#) [App Catalog](#) [Service Keys](#)
2. Under [Service Keys](#), you can view current service keys that are available for third-party API access. Use [Search](#) to search for a subset of destinations.
 - Alias
 - API Key
 - Roles
 - URL
 - Actions
3. Select [+](#) to add a service key.
 1. In [Add Service Key](#), create a service key.

Service Key Properties

Property	Description
Alias	The alias associated with an API key.
Roles	Select the roles that apply: <ul style="list-style-type: none">• list_published_artifacts – retrieves the list of OData entity sets for the artifact's metadata destination.• download_published_artifacts – retrieves the meta-data for the published artifacts using a service key that is assigned a "list_published_artifacts" role.

2. Select [OK](#) to create a service key.

1.5.5.5.13 Managing Cloud Build Service Settings

(Not available in all COUNTRY/REGIONS) As an administrator, you can manage cloud build signing profiles, and purge cloud build artifacts at the tenant level.

[Create a Signing Profile \[page 277\]](#)

As an administrator, you can create signing profiles for iOS and Android applications.

[Purge Cloud Build Artifacts \[page 279\]](#)

Set up a schedule for purging cloud build logs and build output.

[Enable Detailed Event Logging \[page 280\]](#)

Enable or disable detailed event logging for the Cloud Build standalone service on Mobile Services cockpit.

Related Information

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

1.5.5.5.13.1 Create a Signing Profile

As an administrator, you can create signing profiles for iOS and Android applications.

Context

In order to build and run a binary on Android and iOS, the binaries must be signed. As an administrator you can manage your enterprise signing profiles, which are used by the cloud build service to build the SAP Fiori Client IPA and APK files. You can generate a new Android signing profile, and you can upload an existing Android or iOS signing profile.

Procedure

1. In Mobile Services cockpit, select [Settings](#) > [Cloud Build](#) > [Signing Profiles](#) >
2. View the existing signing profiles already generated or uploaded for Cloud Build Service [the total number of signing profiles is indicated after [Signing Profiles \(#\)](#)]. You can search for specific profiles, or sort the existing profiles to see a subset of the total signing profiles. The profiles are presented by platform, including Android and iOS.

Signing Profiles

Column	Description
Profile Name	The name assigned to the signing profile
Created By	The identifier of the person who created the signing profile.
Expiry Date	The date that the signing profile expires, in local or UTC time zone.
Profile Type	The profile type, including Uploaded and Generated.
Actions	The actions you can take, such as delete or edit.

3. As administrator you can generate a new signing profile for Android, or upload an existing one for both iOS and Android.

If you select [Generate](#) to generate a new Android signing profile, provide the following information:

- **Required information:**
 - Profile Name
 - Validity (in years)
 - Common Name (current user's name)
- **Optional information:**
 - Organization
 - Organization Unit
 - City or Locality
 - State or Province
 - Country Code

If you select [Upload](#) to upload an existing iOS or Android signing profile, provide the following information:

- **For the iOS platform:**
 - Profile Name
 - Signing Certificate (in .p12 format)
 - Private Key Passphrase
 - Provisioning Profile (in .mobileprovisioning format)
- **For the Android platform:**
 - Profile Name
 - KeyStore File (in .keystore or .jks format)
 - KeyStore Password
 - Key Alias

- Key Password
4. You can edit an existing uploaded signing profile for Android or iOS platforms.

Select the profile you want to edit and make the required changes. You cannot change the name of the signing profile.

Note

You cannot edit generated Android signing profiles.

5. You can delete an existing uploaded signing profile for Android or iOS platforms.
6. Save the new signing profile.
The certificate expiration date appears for each profile on the list.


1.5.5.5.13.2 Purge Cloud Build Artifacts

Set up a schedule for purging cloud build logs and build output.

Context

Purging is at the tenant level. Cloud build logs are kept for the number of days that you specify, and purged when the scheduler task runs. Output and artifacts are kept for the number of builds that you specify, and purged when the scheduler task runs.

Procedure

1. In Mobile Services cockpit, select [Settings](#) > [Cloud Build](#) > [Purge Settings](#).
2. If [Enable Purge Settings](#) is disabled, select .
3. Select [Enable Purge Settings](#) and edit the purge settings.

Purge Settings

Settings	Description
Clear logs of successful builds after	Identify when to clear logs for successful builds – Immediately, or after 1, 3, 7, or 30 days. Default is 7 days.
Clear logs of unsuccessful builds after	Identify when to clear logs for unsuccessful builds – Immediately, or after 1, 3, 7, or 30 days. Default is 7 days.
Keep output of last n successful builds per app	Identify how long to keep the artifacts from the last 1-5 successful app builds. Default is 3 successful builds.
Keep output of last n unsuccessful builds per app	Identify how long to keep the artifacts from the last 1-5 unsuccessful app builds. Default is 3 unsuccessful builds.

Settings	Description
Last purge	Shows the last successful purge in the YYYY-MM-DD HH:MM:SS format (local or UTC time zone).

4. Select [Save](#).

Next Steps

The data is kept until the scheduler task runs, typically every 12 hours, unless otherwise configured on the server side. A build with a PENDING status will not be effected by the purge settings.

1.5.5.5.13.3 Enable Detailed Event Logging

Enable or disable detailed event logging for the Cloud Build standalone service on Mobile Services cockpit.

Context

When disabled, error logging continues but the detailed create, read, update, and delete operations are not included.

Procedure

1. In Mobile Services cockpit, select [Settings](#) [Cloud Build](#)
2. Select [Enable Detailed Event Log](#) in the upper right corner to start logging the initialization of default purge settings for cloud build. To see the detailed event log related to build process for the cloud build, you need to enable the detailed event log for cloud build feature at the application level (see [Viewing Event Logs](#)).
3. At any time, select [Disable Detailed Event Log](#) to stop logging these operations. Error logging continues.

Related Information

[Viewing Event Logs \[page 237\]](#)

1.5.6 Security

Describes basic security features for SAP Mobile Services in the Cloud Foundry environment.

[Configuring App Security \[page 282\]](#)

Configure security at the application level for the Cloud Foundry environment.

[Configuring and Testing Authorized Access \[page 292\]](#)

This procedure allows you to verify that the client has permission to access a Mobile Service application in the Cloud Foundry Environment.

[Configuring Security Trust \[page 294\]](#)

In Mobile Services, trust must be established between Mobile Services and your SAP BTP sub-account in the Cloud Foundry environment when using some security types or features.

[Configuring Cross Context SSO \[page 295\]](#)

Configure cross context SSO so that identity is transferred from a web app to a mobile app during onboarding. Depending on how the feature is implemented, this enables a user to login and configure an app from the desktop or an Android or iOS mobile phone.

[Configuring IAS Security \[page 297\]](#)

When a mobile application is created with Identity Authentication service (IAS) support, a service instance is created and bound to the mobile application.

[Configuring Digitally-Signed QR Codes \[page 299\]](#)

Configure a digitally-signed QR code that enables users to scan a QR code and onboard securely to a mobile services application.

[Revoking OAuth Tokens \[page 301\]](#)

Revoke an OAuth token to force a registered user to login using a new OAuth token.

[Session Cookies \[page 302\]](#)

Session cookies are used to maintain sessions between mobile app clients and Mobile Services.

[Service Keys \[page 303\]](#)

For some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials.

[SAP Data Custodian Key Management Service Integration \(Restricted Availability\) \[page 307\]](#)

An option is available for some customers to manage their own keys in the Cloud Foundry environment, rather than have SAP manage them.

[Configuring Android Attestation \[page 308\]](#)

(Native/MDK only) Device attestation enables developers and administrators to learn about the software and hardware environment of devices that are trying to connect to enterprise apps and workspaces.

[Configuring iOS Attestation \[page 312\]](#)

(Native/MDK only) Device attestation enables developers and administrators to learn about the software and hardware environment of devices that are trying to connect to enterprise apps and workspaces.

[Data Protection and Privacy \[page 315\]](#)

SAP Mobile Services does not track or store personal data, but tracks data that is related to the mobile services and set-up details.

1.5.6.1 Configuring App Security

Configure security at the application level for the Cloud Foundry environment.

Context

When you define a new application in the Cloud Foundry environment, an OAuth client is created automatically by the server. A non-null default value is set for client ID and the redirect URL once the application is created.

If you select [OAuth](#) or [SAML](#) for app security authentication, API Key is not enabled. If API Key is configured as part of SDK bootstrapping, services such as setting exchange, log upload, and usage upload are accessed before Mobile Services authentication.

If you select [API Key Only](#) for app security authentication, and enable API Key, anonymous access is used. This is similar to the "No auth" option that is available in the Neo landscape. This combination also provides access to client resources. If you select [API Key Only](#), you must enable API Key. The `x-smp-deviceid` header is mandatory when using API Key. You can use an arbitrary value for the device ID when testing using Postman.

To increase application security, you can provide a specific list or a range of [Allowed IP/CIDR](#) addresses and notations. Requests for the allowed list are performed; requests for addresses or notations that are not specified are ignored. If no entries are provided, there is no restriction, and requests are performed for all addresses.

Security Recommendations

Keep in mind these recommendations when configuring app security:


- Apps should enable the passcode policy.
- Apps should primarily attempt to use the OAuth security configuration.
- Service Keys should favor the X.509 type.
- Review the Audit Logs regularly.
- For critical iOS and Android applications, consider implementing the attestation feature for enhanced security.
- Certificates should be renewed regularly and before they expire to avoid any expiration issues.

For more details see [SAP BTP Security Recommendations](#) and filter on the Mobile Services component.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, and [Security](#). View the current security settings, or select the edit button to make changes to a section.
3. (Optional) Lock the application so others can't make changes to the same application.
4. Under [Application Settings](#), view or edit settings.
 1. (Web applications only) Select [CSRF Protection](#)

2. For [Security Configuration](#), select a security option:
 - [API Key Only](#) – uses an API key for authentication. The key is generated or you can select additional keys in a later step. No other authentication method can be used with this choice.
 - [Basic](#) – enables stand-alone basic authentication either with or without Cloud Connector. You must download and import a Trust Configuration into your cloud sub-account as described in [Configuring Security Trust](#).
 - (Does not appear if IAS is selected) [Basic with SAP ID Service](#) – authentication delegates to the SAP ID service. If your SAP BTP account uses a different identity provider, this choice fails.
 - [OAuth](#) – an open protocol for securely authorizing applications using a standard method. SAP BTP must be the authorization server. This is the default option, and the OAuth client is generated automatically. You can use API keys with this choice.
The Proof Key for Code Exchange (PKCE), by OAuth Public Clients, is supported for Authorization Code Grant when using OAuth authentication type. The app must implement the OAuth APIs used by the SDK.
 - [SAML](#) – uses SAML 2.0 supplied by SAP BTP (the SAP ID service authenticates users against both SAP user accounts and SCN accounts). You can use API keys with this choice.
 - [HTTP](#) – enables you to use an HTTP URL to access OAuth endpoints by passing a valid bearer token to the back end for authentication. This is similar to the HTTP module available in SAP Mobile Platform, and is especially useful when migrating applications from SAP Mobile Platform to Mobile Services. You must download and import a Trust Configuration into your cloud sub-account as described in [Configuring Security Trust](#).
3. If you select [OAuth](#) as the security configuration (the default), select the edit button to configure application settings. Click [OK](#) to save changes.

In [Edit Application Settings](#), under [OAuth Clients](#), click the  icon. You can add or remove OAuth clients or regenerate redirect URLs.

OAuth Client Settings

Field	Description
Client ID	To regenerate the client ID, select an ID and then Regenerate . The ID is automatically generated, and identifies the application client to the authorization server.
Redirect URL	The redirect URL for the OAuth client.

Under [OAuth APIs](#) you can view and copy the URLs for [OAuth Authorization](#) and [OAuth Token](#).

Under [Cross Context SSO](#), you can enable identity transfer for SSO. See [Configuring Cross Context SSO](#) for information about setting up identity transfer for use in application onboarding.

Cross Context SSO Settings

Field	Description
Enable Cross Context SSO	Select this option to enable cross context SSO. Additional fields appear.
SSO Token Timeout	How long before the temporary login token expires in seconds. The default is 30 seconds.

Field	Description
SSO Onboarding URL	(Read/copy only) The onboarding URL.
SSO Onboarding URL for Android Widget Support (Does not appear if IAS is selected)	(Read/copy only) The onboarding URL that supports the Android Widget when it is enabled. You can make this link available to users.


Note


IAS-based apps do not support read-only tokens. When you configure security for IAS-based apps and enable Cross Context SSO, the "SSO Onboarding URL for Android Widget Support" option does not appear.

Under [QR Code Signature Settings](#) you can enable use of digital QR code signatures for onboarding. See [Configuring Digitally-Signed QR Codes](#) for information about setting up QR code signature settings for onboarding users.

QR Code Signature Settings

Field	Description
Public Key	The public key is used on the client side to verify the digital signature.
Private Key	The private key is used on the server side to sign the QR Code, and is not exposed to any client.


(Mobile Development Kit only) Once the Public Key appears, you can select  to download a file containing the key. This is useful when working with a developer or to keep a back-up copy.

(Mobile Development Kit only) You can also select  to copy the public key to the clipboard, and then paste the contents elsewhere.

- If you select [Basic with SAP ID Service](#), [Basic](#), or [SAML](#) instead of [OAuth](#), you are notified that the change affects the application availability for the device user, and that an application restart is required to update the application's configuration. Select **OK**.

If you select [Basic](#), provide the [Basic Settings](#), and select **OK**. Also, the [Mobile Connectivity](#) feature is assigned automatically to the application to provide a proxy service, and cannot be removed (navigate to the [Info](#) tab to verify that [Mobile Connectivity](#) appears under [Assigned Features](#)).

Basic Settings

Field	Description
URL	The URL (endpoint) to use for basic authentication. Mobile Services uses the URL to authenticate mobile apps by sending an incorrect test user name/password to the customer-configured endpoint. The endpoint must return a 401 (Unauthorized) response message to demonstrate that it can reject bad credentials that are sent using <code>Authorization: Basic</code> header. This test is done according to RFC 2617 (as described in: https://tools.ietf.org/html/rfc2617 ).

Field	Description
Use Cloud Connector	Enables you to use cloud connector.
Cloud Connector Location ID	Your cloud connector identifier.

- Once you select [HTTP](#), if you check [SAP Destination service](#), you must create a destination in SAP BTP Cockpit first, and enter the Cloud Destination that you just created. Or, you can enter an [URL](#) directly. Then input the [Response Header for User Name Extraction](#) fields, and select [OK](#) to save and confirm. The client needs to attach a valid bearer token in the header to access Mobile Service endpoints. The bearer token may be acquired from another Identity Provider. Mobile Services then sends the bearer token to the configured Cloud Destination or URL for validation. Once the token is validated using the "URL" provided, Mobile Services uses the [Response Header for User Name Extraction](#) field value as the header name to retrieve the user name from the response.

HTTP Settings

Field	Description
SAP Destination service	Select to use a cloud destination to validate the given bearer token. You must provide the Cloud Destination Name.
Cloud Destination Name	You must first configure a cloud destination in the SAP BTP cockpit. Access your subaccount, and then navigate to Connectivity > Destinations to create a destination, as described in Managing Destinations . The destination's URL should also meet the requirement described in the URL field. Then input the destination name to this field.
Response Header for User Name Extraction	The header name that Mobile Services uses to extract the user name from the URL response, when it passes the bearer token validation.
URL	Enter the URL to use to validate the given bearer token. This URL must support the GET method, and be able to extract the "Authorization" header as the bearer token for verification. If the token passes validation, this URL must respond with 200 HTTP status, while attaching the user's name to the response header. If the token fails validation, this URL must respond with 401 HTTP status.
Use Cloud Connector	Enable to use a cloud connector.
Cloud Connector Location ID	The cloud connector identifier.
Enable User Attributes	When enabled, the app can access additional attributes for the current user, other than userID, such as email address, groups, addresses, and so forth. The attributes must be in a specific format. For more information, see Enabling User Attributes in the paragraph that follows.

Enabling User Attributes

When you authenticate with an Identity provider, it might provide additional attributes for the current logged-in user other than userID, like email, groups, address etc. For SAML 2.0, the attributes are included in the assertion as one attribute statement with multiple values in it. For HTTP authentication, Mobile Services allows HTTP endpoints to provide user attributes via this option. If you select [Enable User Attributes](#), then Mobile Services parsed the URL or destination's response body in JSON format, and propagate these attributes to the SAML attributes while generating the Mobile Service internal SAML assertion.

Here is an example of the response:

```
{
  "email" : "someone@sap.com",
  "group": "[ 'Group1', 'Group2' ]",
  "department": "IT"
}
```

With this function, you can map different user attributes to different roles to achieve the access control. The configuration step as following:

1. In Mobile Services cockpit, create a role. To do so, navigate to ► [Mobile Application](#) ► [Native/MDK](#) ►, select the application you are using, then click the [Security](#) tab. In the [Role Settings](#) section, click the [Edit](#) button, check [Enable Role Settings](#), enter a role name, and then save it.
2. In SAP BTP cockpit, create a role collection. To do so, go to your sub-account, navigate to ► [Security](#) ► [Role Collection](#) ►, click the [Create](#) button, give the role collection a name, and then click the [Edit](#) button of this role collection to edit the following information:
 - [Roles](#): select the Role Name and Template Name with the Role you created in the first step, and select the current app's Application Identifier.
 - [Attribute Mappings](#): select the Identity provider for Mobile Services (see [Configuring Security Trust](#) for information). Then enter the "Attribute" and "Value" according to the URL or Destination's response body's key and value. If current user's attribute and value matches with the role collection's definition, the corresponding role will be assigned to the user while login.
5. Under [XSUAA Settings](#), view or edit extended services for UAA (XSUAA). See [Demystifying XSUAA in SAP Cloud Foundry](#) for additional information.

If the [XSUAA Settings](#) section does not appear, either the feature is not supported for your configuration or the [IAS Settings](#) option was selected instead. If the edit button does not appear, then values are read-only. When this option is selected, all services, such as registration, are protected with CSRF tokens. Proxied endpoints aren't protected, since they are likely to be protected on the back end..

Note

You cannot migrate existing mobile apps that are configured with XSUAA to IAS. IAS is only supported for new mobile apps. You must recreate the mobile app and configure the new app with IAS settings.

XSUAA Settings

Field	Description
XSUAA Service	<p>Select Default Instance or an existing service instance from the list.</p> <p>Using an existing XSUAA service instance is useful if you already have a Cloud Foundry application deployed and want to connect to this application from Mobile Services.</p> <p>Select the default instance for other scenarios. Note that if you select the default, you can select OAuth Settings.</p>

Field	Description
xs-security.json	Select the JSON file that defines the authentication methods and authorization types used to access your application. See Application Security Descriptor Configuration Syntax for additional information.
Token Lifetime	<p>Enter the lifetime for a token in number of days, hours, or minutes (12 hours is the default). The entry is subject to any limitations imposed by the server. The maximum token lifetime is 86400 seconds - 24 hours.</p> <p>If user credentials are changed or the user account is disabled, the OAuth tokens are automatically revoked, and the user is not able to access mobile application content.</p>
Refresh Token Lifetime	Enter the maximum lifetime for a refresh token in number of days, hours, or minutes (7 days is the default). The entry is subject to any limitations imposed by the server. The maximum refresh token lifetime is 31536000 seconds - 365 days.
Approved Providers	<p>Indicate the approved identity providers for the application:</p> <ul style="list-style-type: none"> • All (default) – all identity providers are approved for the app. • Selected Providers – enter one or more approved identity providers for the app (hit Enter after each entry). <div data-bbox="890 1182 1402 1400" data-label="Complex-Block"> <p>Note</p> <p>Since Mobile Services cockpit cannot validate the identity provider entries, it is very important to enter them correctly. Otherwise, users will not be able to log in successfully.</p> </div> <p>The Selected Providers option enables you to be specific about the identity providers for the app, especially if you have multiple providers configured for the subaccount.</p> <p>For example, say you have two identity providers configured for the subaccount – "sap.default" (origin of default identity provider "SAP ID Service") and "hanamobile_sci". Assume you have two mobile applications, and one application needs to be protected by the "SAP ID Service", and another one by "hanamobile_sci". You can configure "sap.default" as the approved provider for the first application, and "hanamobile_sci" for second application.</p>

Field	Description
System Attributes in Token	<p>The <code>system-attributes</code> property controls which properties should be included in the JWT token generated by an xsuaa service instance. Its purpose is to decrease the size of JWT by excluding some properties, especially when many groups have been assigned to a user in IDP.</p> <p>You can configure <code>system-attributes</code> for a mobile applications xsuaa service instance using:</p> <ul style="list-style-type: none"> • <code>Rolecollections</code> (default for new applications) • <code>Groups</code> <p>For legacy applications (those that were created before the introduction of this feature, and do not have this property set), the value of <code>system-attributes</code> might be:</p> <ul style="list-style-type: none"> • If <code>system-attributes</code> was set in <code>xs-security.json</code>, retrieve the property value and use it as the <code>system-attributes</code> value. • If not, use the <code>Groups</code> or <code>Rolecollections</code> value as the <code>system-attributes</code> value. <p>This option is only available for OAuth. To learn more, see Application Security Descriptor Configuration Syntax.</p>

- Under [IAS Settings](#), view or edit SAP Identity Authentication service settings.

See *Configuring IAS Security* for information about setting up IAS integration support.


Note

You cannot migrate existing mobile apps that are configured with XSUAA to IAS. IAS is only supported for new mobile apps.

- (Does not appear if IAS is selected) Under [Role Settings](#), view or edit settings. In [Edit Role Settings](#), select [Enable Role Settings](#) to use roles.

Once enabled, you can specify one or more roles, such as Admin or Developer, and click *OK*. If you make any changes, you must restart the application.

- Under [Application Versioning](#), select [Enable Application Versioning](#) to manage versions for this application. Once enabled, you can view and edit versions.


- Click the  icon to add a version.
- Modify the application version settings:

Version Settings


Field	Description
Active	Whether the application version is active and available for use.

Field	Description
Version	Application version, such as 1 . 0, 3 . 3 PL02, and 5 . 2. A version is mandatory.
Check Platforms	Indicates whether the Mobile Services server should check for specific platforms when processing app versions. Set the toggle to <i>Off</i> (default) if the app versions for all platforms should be processed, and set to <i>On</i> if app versions for specific platforms should be processed. Once set to <i>On</i> , select one or more platforms from the list of <i>iOS</i> , <i>Android</i> , and <i>Windows</i> . Note that apps that were introduced before <i>Check Platforms</i> was introduced (Version: 2303) are set to <i>Off</i> .
Description	Application version description, such as BETA, Initial release, and Patch to Version 3.3 - Windows only.
Actions	Actions you can perform, such as edit or delete.


3. Click *OK* to save, or *Cancel*. If you make any changes, you must restart the application.

9. Under *API Key*, if you selected *OAuth*, *SAML*, or *API Key Only* as the Security Configuration, you can view or edit API keys. If you selected *API Key Only*, an API key is created using the current date and time. You must have at least one API key or you receive a warning. Select  to add or delete keys.

Under *Anonymous Access*, view or edit settings. In *Anonymous Access*, select *Allow Anonymous Access* to enable client-side applications to use API keys as credentials.

1. To edit, click the  icon to add an API key.
2. Add one or more valid API keys for the application.

API Key Settings

Field	Description
API Key	When implemented, at least one API key is required.
Creation Date	The server generates the creation date once the API key is created.
Actions	Select  to delete an API key.

3. Click *OK* to save, or *Cancel*. The API keys and their creation dates appear. If you make any changes, you must restart the application.

10. Under *Allowed IP/CIDR*, view the list of allowed Internet Protocol/Classless Inter-Domain Routing (IP/CIDR) addresses, or ranges of addresses. Any request that includes one of these addresses will be performed; requests for addresses that are not in the list will be ignored. You can add, change, and delete entries.

To make entries:

1. Select the edit icon.
 2. In [Edit Allowed IP/CIDR](#), select the plus icon, and then enter an address or range of addresses, for example:
 - 10.0.0.3
 - 10.100.0.1/32
 - 2a0f:ed40::/29
 3. To delete an entry, select [X](#).
 4. To modify an entry, make changes to a current entry.
 5. Select [OK](#) to save changes, and confirm. The change appears in the list.
11. Under [Cross Domain Access](#), view the list of cross-domain access entries. These entries enable application users to access additional security domains based on the values you provide. You can add, change, and delete entries. See also, *Cross-Origin Resource Sharing Requests*.

To make entries:

1. Select the [+](#) icon to add an entry.
2. In [Create Cross Domain Access](#), enter the values, and.

Cross Domain Access Properties

Properties	Descriptions
URI Pattern	A string of characters that matches the destination.
Origin	<p>A comma-delimited list of URIs that can access the resource, for example, <code>http://example.com</code> or <code>http://*.example.com</code>.</p> <p>The default value "*" means any URI can access the resource.</p> <p>An empty value prevents cross-origin resource sharing; only URIs in the same origin can access the resource.</p>
Methods	A comma-delimited list of HTTP methods (such as <code>GET</code> and <code>DELETE</code>) that are allowed when accessing the resource.
Max Age	The number of seconds for which the results of a request can be cached. The default is 3600 seconds (60 minutes).
Allow Credential	Always set to On. The server includes cookies when it submits requests.

Properties	Descriptions
Headers	<p>A comma-delimited list of HTTP request headers that you can specify in requests.</p> <div> <p>Note</p> <p>An empty value means any requested headers are accepted.</p> </div> <p>The following headers are usually used in requesting Mobile Services:</p> <ul style="list-style-type: none"> • <code>accept</code> • <code>authorization</code> • <code>maxdataserviceversion</code> • <code>x-smp-appcid</code> • <code>x-smp-appid</code>
Expose Headers	A comma-delimited list of response headers that browsers can access.

3. Select [Save](#). The new entry appears in the list.

12. If you locked the application while making changes, select [Unlock](#) to release it.

Related Information

[Configuring Applications \[page 45\]](#)

[Defining Applications \(Cloud Foundry\) \[page 46\]](#)

[Service Plans \[page 23\]](#)

[Cross-Origin Resource Sharing Requests \[page 443\]](#)

[Configuring Security Trust \[page 294\]](#)

[Configuring Digitally-Signed QR Codes \[page 299\]](#)

[Configuring Cross Context SSO \[page 295\]](#)

[Configuring IAS Security \[page 297\]](#)

1.5.6.2 Configuring and Testing Authorized Access

This procedure allows you to verify that the client has permission to access a Mobile Service application in the Cloud Foundry Environment.

Context

This procedure provides a step-by-step guide about how to configure the application environment and run a simple test case.

For applications based on XSUAA, setup consists of these tasks:

1. In the Mobile Services cockpit, set up authorized client access for the application.
2. In the SAP BTP cockpit, set up Role Collections and mapping to IDP groups.
3. Set up the user and related groups in the identity provider (IDP), and then log in to SAP Mobile Services to get the access token.
4. Send a request to the application to test that the access rule is satisfied via the 200 OK or 403 Forbidden responses.

For applications based on IAS, assign users to the group with the name that starts with "SAPMS_" and ends with the role name that is configured for the application in the IAS tenant. For example, if you configure the "mobile_user" role for the application, then the customer must create the "SAPMS_mobile_user" group in their IAS tenant, and assign the group to all users who can access the mobile services application.

Procedure

1. In the Mobile Services cockpit, set up authorized user access to the application:
 1. Log in to the Mobile Services cockpit.
 2. Select an existing application or create a new one.
 3. On the [Security](#) tab, you can set up the security configuration properties, including Application Versioning and Role Settings.
 4. Click [Edit](#) for [Application Versioning](#) and enable Application Versioning. When Versioning is enabled, add one or more Application Versioning items (v1.0 and V1.1, for example, where v1.1 is the active version) and click [Save](#) to save the configuration.
 5. Click [Edit](#) for [Role Settings](#) and enable role settings and add roles. Add role1, role2, role3, etc. and click [OK](#) to save.
2. In the SAP BTP cockpit, set up the Role Collections and mapping to IDP groups:
 1. From the [Security](#) menu in the target tenant setting page, select [Role Collections](#), and click [New Role Collection](#) to create a new Role Collection for the application.
 2. Fill in the required fields, including a role collection name (such as, role_collection_app1), and click [Save](#).
The Role Collection is created.
 3. Click the link for the Role Collection name to open the Role Collections page. To add a role, click [Add Role](#).

4. In the [Add Role](#) dialog, select the [Application Identifier](#) (the XSUAA client ID listed on the [Application Detail](#) page).
 5. For [Role Template](#), select [role1](#).
 6. For [Role](#), enter [role1](#) and click [Save](#).
 7. Repeat the previous three steps for [role2](#) and [role3](#).
 8. Return to the [Security](#) menu and choose [Trust Configuration](#) to set up the mapping between the Role Collection and the Group in IDP configuration.
 9. Click the link in the [Name](#) column (xsuaa-monitoring-idp, for example).
 10. Click [Role Collection Mappings](#) to set up the mappings.
 11. Click [Edit](#) to create a new Role Collection Mapping.
 12. Choose the name of the Role Collection you created (role_collection_app1, for example) and enter the group name in the [value](#) field (group1, for example). The group name must already exist in the IDP settings.
 13. Click [Save](#).
3. Set up the user and related groups in IDP, then log in to Mobile Services to get the Access Token:
 1. Use a utility such as SoapUI to verify authorized user access to the application. Firstly, send a specific request to the Mock IDP "xsuaa-monitoring-idp" to generate a specific Access Token. Set [username](#) to "testuser" and [group](#) to "group1". Also set the related URLs and Client ID.
 2. Generate the Access Token.
The Access Token is generated and it includes role1, role2, and role3.
 3. Change the group to group2 and generate a new Access Token.
The Access Token is generated and it does **not** include role1, role2, and role3.
 4. Send a request to the application to test that the access rule is satisfied via the 200 OK or 403 Forbidden responses:
 1. On the [Native/MDK](#) page, disable the [Application Versioning](#) check and leave the [Role Settings](#) check enabled.
 2. Send a request to the application using the Access Token that includes any of role1, role2, and role3.
The response code is 201 OK, indicating that the request passed the authorized user access check and the app can be onboarded successfully.
 3. Send a request to the application using the Access Token that does **not** include any of role1, role2, and role3.
The response code is 403 Forbidden, indicating that the request was rejected by the authorized user access check.
 4. On the [Native/MDK](#) page, enable the [Application Versioning](#) check and disable the [Role Settings](#) check.
 5. Set the header "X-APP-VERSION" value to a valid Application Version (such as, the active version of the app, v1.1) and send the request
The response code is 200 OK, indicating that the request passed the authorized user access check.
 6. Set the header "X-APP-VERSION" value to an invalid Application Version (such as, the inactive version of the app, v1.0) and send the request
The response code is 403 Forbidden, indicating that the request was rejected by the authorized user access check.
 7. On the [Native/MDK](#) page, disable both the [Application Versioning](#) check and the [Role Settings](#) check.
 8. Send a request without any roles, and also with an invalid Application Version (such as, the inactive version of the app, v1.0), and send the request to the application.
The response code is 200 OK, indicating that no version or role check was performed and the request passed the authorized user access check.

1.5.6.3 Configuring Security Trust

In Mobile Services, trust must be established between Mobile Services and your SAP BTP sub-account in the Cloud Foundry environment when using some security types or features.

Prerequisites

Default and custom trust configuration must already be established in SAP BTP cockpit.

For information about establishing trust, see [Establish Trust and Federation with UAA Using Any SAML Identity Provider](#).

Context




Importing trust configuration metadata into your cloud subaccount enables Mobile Services to generate user authentication tokens for users. Importing the trust configuration only needs to be done once. Trust is required for applications that use the following:

- Basic authentication
- HTTP authentication
- API Key (Anonymous) authentication
- Multi-user/device sharing scenarios
- Micro App applications in supported countries/regions (for example, WeChat and DingTalk authentication)

Note

If all configured applications only use OAuth or SAML authentication, you do not need to configure security trust.

Procedure

1. In Mobile Services cockpit, select  [Settings](#)  [Security](#) .
2. Select [Metadata Download](#) to download the trust configuration from Mobile Services cockpit to your local system. The file will be used to establish a trusted relationship with SAP BTP. This only needs to be done once.

In [Download Metadata](#), specify the metadata expiration date, and then select [Download](#). You can select one year (default), or use the date picker to select the month and year. The maximum validity is ten years.

3. From SAP BTP, import the downloaded trust configuration file from the previous step, as a trusted identity provider, using your customer subaccount.

Note

When importing the Identity Provider in SAP BTP, disable the “Available for User Login” checkbox, because the trusted Identity Provider is not a real identity provider and cannot be used for user login.

Disabling the “Available for User Login” checkbox ensures the identity provider does not appear on the XSUAA login page.

4. Select [Test](#) to make sure the metadata has been established successfully as a trusted provider.

If the trust configuration is not established correctly, you may see a message like `SAML metadata might not have been imported as trusted identity provider`.

1.5.6.4 Configuring Cross Context SSO

Configure cross context SSO so that identity is transferred from a web app to a mobile app during onboarding. Depending on how the feature is implemented, this enables a user to login and configure an app from the desktop or an Android or iOS mobile phone.

Prerequisites

- Onboarding SSO must be set for the app via the SAP BTP SDK for Android and SAP BTP SDK for iOS.
- SAP BTP must be used as the authorization server.
- The app security configuration must be set to [OAuth](#) via Mobile Services cockpit, as described in *Configuring App Security*.
- The feature is supported for Android and iOS apps.
- If users will onboard from an Android mobile phone by clicking a button, you must configure Android Application Links in Mobile Services cockpit through the [Application Links](#) tab, as described in *Creating Application Links*.
- The user experience for onboarding from an iOS device, when clicking a button, will depend on whether Apple Universal Links or custom URL scheme is configured for your app. If both are configured for your app then custom URL scheme is used to provide better user experience.

Apple Universal Links have the effect that the app does not open automatically when a user browses a website in Safari and taps a universal link in the same domain. Instead, iOS opens the link in Safari, expecting the user to continue within the browser, and the user must actively tap the banner to open the app.

Custom URL link avoids this extra step, which provides a better user experience. However, if custom URL schemes are not maintained by an administrator, then the Apple Universal Link technique is used. Both the custom URL scheme and Apple Universal links can be configured in Mobile Services cockpit through the [Application Links](#) tab, as described in *Creating Application Links*.

Context

If cross context SSO is set via the SDK and configured through Mobile Services cockpit, users can onboard in these ways:

- From the desktop: The user opens the onboarding page in a browser, and scans a QR code using the phone's camera. The QR code provides information that can be interpreted by the SAP mobile app.



- From an Android phone: The user opens the onboarding page using a Chrome browser, which is the only supported browser. The page includes a button to click to start the onboard process. When the user selects the button, the Android mobile phone calls the SAP app.
- From an iOS phone: The user opens the onboarding page using a Safari browser, which is the only supported browser. The page includes a button to click to start the onboard process. When the user selects the button, the iOS mobile phone calls the SAP app.

In all cases, a short-term token is used to pass authentication to the app. If the token times out, the user is offered additional chances to onboard using a new short-term token each time.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, and **Security**. View the current security settings, or select the edit button to make changes.
3. Under **Application Settings**, view or edit settings. Select the edit button to configure application settings. The **Edit Application Settings** window appears.
 1. Under **Cross Context SSO**, select **Enable Cross Context SSO**. Additional fields appear.
 2. Configure cross-context SSO settings.


Cross Context SSO Settings

Field	Description
SSO Token Timeout	Specifies how long before the temporary login token expires in seconds. The maximum is 300 seconds, and the default is 30 seconds.
SSO Onboarding URL	(Read/copy only) The onboarding URL. You can make this link available to users. To obtain the passcode: <ol style="list-style-type: none"> 1. Select the copy icon  to copy the SSO Onboarding URL to the clipboard. 2. In a new browser window, paste the URL and then append with <code>#/passcode</code>. 3. When the Temporary Authentication Code screen appears, select the copy icon  to copy the authentication code to the clipboard. 4. Use this value as the passcode to authenticate.

Field	Description
SSO Onboarding URL for Android Widget Support (Does not appear if IAS is selected)	(Read/copy only) The onboarding URL that supports the Android Widget when it is enabled. You can make this link available to users.

Note

IAS-based apps do not support read-only tokens. When you configure security for IAS-based apps and enable Cross Context SSO, the "SSO Onboarding URL for Android Widget Support" option does not appear.

Select the copy icon  to copy the link to the clipboard.

- Click [OK](#) to save changes.

Next Steps

When users click on the onboarding URL or scan the QR code, they are onboarded. If the security code expires before the user can be onboarded, the user is offered the chance to [Get New Code](#).

Related Information

[Configuring App Security \[page 282\]](#)

[Creating Application Links \[page 216\]](#)

1.5.6.5 Configuring IAS Security

When a mobile application is created with Identity Authentication service (IAS) support, a service instance is created and bound to the mobile application.

Prerequisites

Identity Authentication is available to subaccount spaces, only if trust and federation has been established between the SAP BTP subaccount and the Identity Authentication tenant. For more information see [Establish Trust and Federation Between UAA and Identity Authentication](#) and [Getting Started with the Identity Service of SAP BTP](#). If trust and federation has not been established between the Identity Authentication tenant and the SAP BTP subaccount, only XSUAA-based mobile applications can be created in subaccount spaces.

When creating or updating IAS-based mobile applications on Mobile Services cockpit, you can upload a customized security configuration to use as the default Identity Authentication service instance. For configuration syntax see [Reference Information for the Identity Service of SAP BTP](#).

Context

Mobile Services supports protecting mobile application with your SAP Cloud Identity Services - Identity Authentication tenant using Open ID Connect (OIDC). When a mobile application is created with IAS support, a service instance for the Identity Service of SAP BTP is created and bound to the mobile application. This redirects mobile application users to the IAS tenant upon login, and generates an IAS token from the IAS tenant after login completes. Mobile clients also use IAS tokens to access the mobile application, and use the IAS token to propagate the user to the back-end server.

The IAS token is an identification token, and does not include authorization information (scopes or roles). Mobile Services supports using the "groups" property to grant authorization to users. Mobile Services treats those groups that start with "SAPMS_" as roles, and the role names are the values that remain when the "SAPMS_" prefix is removed.

For example, a user might create an "SAPMS_Administrator" group in the IAS tenant, and assign users to the group. The users are granted access to APIs that require the "Administrator" role in Mobile Services. Mobile Services supports the "Administrator" and "Helpdesk" roles that enable runtime users to access the single-application cockpit. The "Administrator" role is a read-write role, and "Helpdesk" is a read-only role.

Note that Mobile Services runtime supports revoking user tokens for apps that use Identity Authentication service (IAS), including Software as a Service (SaaS) apps. The app must implement OAuth authorization.

Also note that you cannot migrate existing mobile apps that are configured with XSUAA to IAS. IAS is only supported for new mobile apps. You must recreate the mobile app and configure the new app with IAS settings.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#).
2. Select an application, and [Security](#). View the current security settings, or select the edit button to make changes.
3. Under [IAS Settings](#), view or edit settings. Select the edit button to configure settings. The [Edit IAS Settings](#) window appears.

For information about IAS Service Token Settings, see [Token Policy Configuration for Applications](#).

IAS Settings

Field	Description
IAS Service	Select Default Instance or an existing service instance from the list.

Field	Description
Security-config.json	Select the JSON file that defines the authentication methods and authorization types used to access your application. See Reference Information for the Identity Service of SAP BTP for additional information.
Token Lifetimes	Enter the lifetime for a token in number of hours or minutes (15 minutes is the default). Values include 10 minutes through 1 hour. The entry is subject to any limitations imposed by the server.
Refresh Token Lifetime	Enter the lifetime for a refresh token in number of days, hours, or minutes (30 days is the default). Values include 1 hour through 180 days. The entry is subject to any limitations imposed by the server.

- Click [OK](#) to save changes.

1.5.6.6 Configuring Digitally-Signed QR Codes

Configure a digitally-signed QR code that enables users to scan a QR code and onboard securely to a mobile services application.

Prerequisites

This feature requires the OAuth security configuration.


Context

The digitally-signed QR code helps prevent a malicious person from gaining access. An administrator configures the public and private key pair used to sign the QR code in JSON Web Signature (JWS) format. Use Mobile Services to generate the keys, or add your own keys.

When the signature feature is enabled:

- The QR code on the [Mobile Application > Native/MDK > <app_name> > APIs tab](#) is signed, including Configuration, Application Link, generic URI scheme, and MDK URI scheme.
- The Cross Context SSO on [Mobile Application > Native/MDK > <app_name> > Security tab](#) is also signed.

You have two ways to manage the key pair: one way is to click [Generate Key Pair](#) to generate a new key pair; another way is to copy and paste your own public and private keys, and upload a private key. Keep in mind:




- The key pair algorithm only supports RSA and EC (Elliptic Curve).
 - For the RSA algorithm, supported bits lengths are 2048, 3072, 4096.
 - For the EC algorithm, supported bits lengths are 256, 384, 521.
- Only the PEM-Encoded PKCS#8 unencrypted key format is supported:
 - Public Key should start with -----BEGIN PUBLIC KEY----- and end with -----END PUBLIC KEY-----.
 - Private Key should start with -----BEGIN PRIVATE KEY----- and end with -----END PRIVATE KEY-----.
- When you use the [Generate Key Pair](#) button, an RSA key pair is generated with a 3072 bits length.
- The Public key and Private key should appear and update in pairs.
- When [Enabled QR Code Signature](#) is not enabled, the Public Key and Private Key values are stored (and not deleted), but they do not appear and are not used in onboarding until they are again enabled.
- You can generate your own key pairs using a third-party tool such as [OpenSSL](#) . Examples:
 - Generate an RSA key pair with OpenSSL tool:

```
openssl genrsa -out privkey.pem 4096
openssl rsa -in privkey.pem -pubout -out public.pub
openssl pkcs8 -topk8 -inform pem -in privkey.pem -outform pem -nocrypt
-out private_pkcs8.pem
```

- Generate an EC key pair with OpenSSL tool:

```
openssl ecparam -genkey -name secp521r1 -noout -out ec512-key-pair.pem
openssl ec -in ec512-key-pair.pem -pubout -out public.pem
openssl pkcs8 -topk8 -inform pem -in ec512-key-pair.pem -outform pem
-nocrypt -out ec512-key-pair_private_pkcs8.pem
```

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) .
2. Select an application, and [Security](#). View the current security settings, or select the edit button to make changes in a section.
3. Under [Application Settings](#), verify that the security configuration type is [OAuth](#).
4. Scroll down to [QR Code Signature Settings](#), and select [Enable QR Code Signature](#). Additional fields appear. Configure the QR code signature settings.

QR Code Settings

Field	Description
Public Key	The public key is used on the client side to verify the digital signature.
Private Key	The private key is used on the server side to sign the QR Code, and is not exposed to any client.

You have two ways to manage the key pair: one way is to click Generate Key Pair to generate a new key pair; another way is to copy and paste your own public and private keys.

5. In *QR Code Signature Settings*, click *Generate Key Pair* to generate a new key pair. The Public Key and Private Key fields are populated with the keys. (Alternatively you can copy and paste your own public key, and upload a private key.)
6. In *Secret*, provide at least eight characters of Secret to generate an encrypted private key. The public key changes, and the private key is downloaded from the server.

Be sure to save the private key in a secure location and remember the secret in case you need to redeploy the app. You can also upload a private key, and are prompted to supply the Secret if the private key is encrypted.

Alternatively, you can copy and paste your own public and private keys. Keep in mind the information and guidelines provided in *Context*.

7. Select *Save* and *OK* to confirm.



The encrypted private key appears as *****.

The public and private keys are saved to the SAP Mobile Services server, and are available to iOS or Android clients to verify authentication during onboarding. When users click on the onboarding URL or scan the QR code, they are onboarded using the keys.

Once you input or generate a new key pair, you must not change it. The key pair has been saved to the device application configuration, and changing it would affect the device application.

Note

For Mobile Development Kit apps, you can download the private key or copy it to the clipboard. The download and copy buttons do not appear for other app types or if the Mobile Development Kit app is locked.

- (Mobile Development Kit only) Once the Public Key appears, you can select  to download a file containing the key. This is useful when working with a developer or to keep a back-up copy.
- Select  to copy the public key to the clipboard, and then paste the contents elsewhere.

1.5.6.7 Revoking OAuth Tokens

Revoke an OAuth token to force a registered user to login using a new OAuth token.

Context

An administrator can revoke an OAuth token for a registered user. This action revokes all OAuth tokens that were issued for the user before the time of revocation. When the user logs in again, a new OAuth token is generated. (Similarly, if you delete a user registration as described in *Managing Registered Users*, all OAuth tokens are revoked. A new OAuth token is generated when the user logs in again.)

If the *Revoke* button does not appear, the feature is not supported for the configuration.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#) or [SAP Mobile Cards](#)
2. Select an application, then select [Mobile Settings Exchange](#) under [Assigned Features](#).
3. (Optional) Under [Registered Users](#), you can narrow the focus of the registered users to view by setting filters, such as [User Name](#), [Registration Time Frame](#), [Device Type](#) or [Email Address](#); and selecting [Go](#). Select [Reset Filter](#) to clear your filter entries.

The filtered list of registered users appears. You can also use Sort and Customize Table Columns to further change the list.

4. Find your target registration user, and under [Actions](#), select [Revoke](#).

In [Confirm Revoke](#), select [OK](#) to confirm that all OAuth tokens registered to the user will be revoked.

5. Check the date and time stamp under [Last Revocation \(UTC+0000\)](#) to confirm the change.

The date and time stamp is updated when you click the [Revoke](#) button, and when the user logs out from a device. When the user logs out from a device, all tokens under the same user name are revoked.

Next Steps

When the registered user logs in to the app, a new OAuth token is generated.

Related Information

[Defining Client Policies \[page 61\]](#)

[Managing Registered Users \[page 219\]](#)

1.5.6.8 Session Cookies

Session cookies are used to maintain sessions between mobile app clients and Mobile Services.

As an SAP BTP service, Mobile Services provides two session cookies that are used to maintain sessions between mobile app clients and Mobile Services:

- `JSESSIONID` – session cookie that is used to maintain a session with Mobile Services.
- `__vcap_id__` – session cookie that is used to provide affinity with SAP BTP Cloud Foundry.

When accessing a back-end server through Mobile Services, the back-end server might also generate cookies (session or non-session). Mobile Services does not directly return the cookies generated by the back-end server to the client.

The back-end server session cookies are stored in Mobile Services. When a subsequent request is received for the same back-end server, Mobile Services loads the stored session cookies and appends them to the forwarded request to the back-end server. If the client sends multiple requests to Mobile Services in the same

session, the back-end server session cookies are included in all subsequent requests, even though the client never receives them.

Each back-end server non-session cookie is wrapped as a new cookie in Mobile Services. The wrapped cookie is returned to the client. The client includes that cookie value in subsequent requests, and Mobile Services unwraps the cookie, and appends the original cookie in the request forwarded to the back-end server.

The format of the wrapped cookie name is:

```
cpmscf proxy SAPCPMS <hash value>
```

The value of the wrapped cookie is the Base64-encoded value of the original cookie returned by the back-end server.

For example, the following `set-cookie` response header is a wrapped cookie:

```
set-cookie:
cpmscf_proxy__SAPCPMS_669C30CF0569E4856F2A88A53197E3268AC5A985=eyJ1YWllIjojQkFDS
0VORF9DT09LSUUilCJ2YXxlZSI6IjEyYmZQlNjc4OSIsInBhdGgiOiIvIiwiaWZlZG9tZWluIjoicHJveHltb2
NrLWl1Ym1sZWRLdilmcm92aWRlcilpbmQtdGVzdC5jZmFwcHMuc2FwLmhhbmEub25kZWl1bmQuYy29tIn0
; Expires=Thu, 20 Nov 2025 12:06:04 GMT; Max-Age=99999999; Domain=test-
app.cfapps.sap.hana.ondemand.com; Path=/
```

1.5.6.9 Service Keys

For some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials.

Prerequisites

The service must be able to support service keys. Currently this includes Mobile App Catalog, Mobile Client Resources, Mobile Client Usage and Feedback, Mobile Push Notification, Mobile Settings Exchange, Mobile Micro App, and Mobile Connectivity.

Context

Service keys provide access to a set of services, either in an API Key or an X.509 manner. That means that a back-end service does not require user credentials, but the service key is limited to a specific use case. For example, you can enable an application to send push notifications using the server key as its credentials, without requiring a user to enter credentials, or you can enable back-end services to call service APIs.

Configure service keys at the provider level or the application level in Mobile Services cockpit. If you do not see the [Service Keys](#) tab for a feature, the option is not available. The service keys for subscriber level and provider level are managed separately.

- Subscriber level - the subscriber level app can manage its own service keys. The administrator can only view/delete service keys that exist in the subscriber level app, but can also accept provider service keys.

The subscriber user can't see the provider service key in the subscriber mobile service cockpit, but the subscriber application can use the provider service key in runtime.

- Provider level - the provider level app can only accept provider service keys.

API Key Manner The API key is recommended for users that are accessing multiple services using the same authentication credentials. You have two options for sending the key to the service, using either the `X-API-Key` header or an Authorization header. See *Next Steps* below for usage.

X.509 Key Manner The X.509 key is recommended for service-to-service communication, enabling customer back-end services to call service APIs. See *Next Steps* below for usage.

A feature flag is available that determines whether a service key is visible in Mobile Services cockpit after its initial creation. Industry best practices recommend that new service keys should be visible upon creation, and thereafter no longer appear. This provides additional security by minimizing the risk that someone can access and use the service key maliciously.

- When the `cpmscf-6405-protection-over-service-key` feature flag is turned on (default), the service key is viewable only upon creation (you can copy the key, and paste it to a secure location). After that the service key is masked.
- When the `cpmscf-6405-protection-over-service-key` feature flag is turned off, the service key remains viewable after creation (this provides a way for customers who rely on being able to retrieve the service key for customer support).

Rotate X.509 service keys before they expire by creating a new one. Once a service key expires, a 403 Forbidden response is returned, and the service cannot be called. Users can't obtain tokens from XSUAA, and they cannot access services without tokens.

Note

To rotate certificates without downtime, create a new service key with a new certificate, and implement the new service key on the back end. When the back end is configured with the new service key, delete the old service key.

Procedure


1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK** or **SAP Mobile Cards**.
2. Select an application, and then select a feature under **Assigned Features**.
3. Select **Service Keys**.

View the list of service keys currently configured for the service instance.

Service Key Properties

Properties	Descriptions
Type	Rows are grouped by service key credential type: API Key and X.509.

Properties	Descriptions
Alias	A human readable name for the service key, from 1-64 characters, and used as the username to authenticate a request. Also used when an audit log record is created. Required.
Key Properties	<p>For API Key, the URL used to access the service instance. The URL is generated, and cannot be edited.</p> <p>For X.509, properties include:</p> <ul style="list-style-type: none"> • Key Length • Validity Lifetime • Start Date • Expiration Date
Roles	The roles that are assigned to this service key. Required.
Actions	<p>You can delete a service key, but you cannot edit one. To change a service key, you must delete it, and then create a new one.</p> <p>You can view details for X.509 service keys.</p>

4. To create a new service key, select .

1. In [Add Service Key](#), configure the service key properties.

Service Key Properties

Properties	Descriptions
Alias	Enter a human readable name for the service key, from 1-64 characters, and used as the username to authenticate a request. Also used when an audit log record is created. Required.
Roles	Enter the roles that are assigned to this service key. Required.
Type	Select API Key or X.509 from the list to specify the service key type.
Type: API Key	If you select API Key, the API-key is sent in the request header and used as the password to authenticate a request. The API-key and URL cannot be edited.

Properties	Descriptions
Type: X.509	<p>If you select X.509, enter the properties:</p> <ul style="list-style-type: none"> Key Length - the key size in bits, for example 2048, 4096. Validity Lifetime - how long the X.509 key is valid, such as 1 Day, 1 Month. Start Date - validity lifetime start date, in YYYY-MM-DD HH:MM(UTC+<xxxx>) format. Expiration Date - validity lifetime end date, in YYYY-MM-DD HH:MM(UTC+<xxxx>) format.

2. Select [OK](#) to save.
3. If the `cpmscf-6405-protection-over-service-key` feature flag is turned on (default), the Add Service Key dialog remains open. The new API Key or X.509 access token appears, along with a message recommending that you to use the copy button to save the value. Once you have saved the value in a secure place, click [OK](#) again.
The new service key is added to the list, and is masked.
4. If the `cpmscf-6405-protection-over-service-key` feature flag is turned off, the new service key is added to the list.
 - The [API Key](#) and [URL](#) are generated from service metadata.
 - The X.509 access token is generated by the customer back-end service using the `uaa.clientid`, `uaa.certurl`, `uaa.certificate` and `uaa.key` properties.

To view details for an X.509 service key, select [Details](#). The properties and generated certificate details appear in JSON format. You can copy or download the contents.

5. To remove a service key, select .

Note

Since you cannot edit service keys, you must delete a service key and configure a new one in order to change a service key.

Next Steps

API Key Manner To send the API key to the service, you can either use the `x-API-Key` header or an Authorization header.

- X-API-Key Header**

To authenticate, send the API-key along with the request in the `x-API-Key` header:

```
x-API-Key <API-Key>
```

- Authorization Header**

To authenticate, send the API-key as Basic authentication in the `Authorization` header. The `<Alias>` is the username and the `<API-Key>` is the password. Both need to be joined with a colon (`<Alias>:<API-Key>`) and Base64 encoded.

```
Authorization Basic [Base64(<Alias>:<API-Key>)]
```

X.509 Key Manner The implementation for X.509 service keys is a little different. In the x509 service key properties retrieved from Mobile Services cockpit, the `uaa` property provides the credentials, which are generated by XSUAA and are used to generate the access token; and the `url` property is the base URL of Mobile Services feature.

The customer back-end service uses the `uaa.clientid`, `uaa.certurl`, `uaa.certificate` and `uaa.key` properties to generate the access token, and then uses the access token to access the back-end API of the Mobile Services feature.

Example request (based on curl):

```
curl --cert cert.pem --key key.pem -XPOST {uaa.certurl}/oauth/token -d
'grant_type=client_credentials&{uaa.clientid}'
```

Where, `cert.pem` includes the value of the `uaa.certificate` property, and `key.pem` includes the value of `uaa.key` property.

Note

The certificate and key string contain new line escape sequences (`\n`), which must be replaced with actual line breaks for your operating system, if used in `cert.pem` and `key.pem`.

The access token value can be obtained from the `access_token` attribute in the response body. When accessing the back-end API of Mobile Services feature, the access token needs to be included as "Authorization" header with "Bearer " prefix.


```
Authorization: Bearer <access_token>
```

1.5.6.10 SAP Data Custodian Key Management Service Integration (Restricted Availability)

An option is available for some customers to manage their own keys in the Cloud Foundry environment, rather than have SAP manage them.

Context


Mobile Services integration with the SAP Data Custodian Key Management Service (DCKMS) provides this option. The Mobile Services environment must meet certain conditions and be approved by SAP. Once approved, some customer actions are required for onboarding.

For more information, see SAP Note [3348581](#)  - SAP Data Custodian Key Management Service Integration for Mobile Services (login required).

1.5.6.11 Configuring Android Attestation

(Native/MDK only) Device attestation enables developers and administrators to learn about the software and hardware environment of devices that are trying to connect to enterprise apps and workspaces.

Prerequisites

- Ensure that the correct version of Google Play services is installed on the user's device.
- Make sure customers can access Google server.
- (SafetyNet only) Obtain a Google API Key. This key is needed to call the SafetyNet Attestation API. To create an API key, see [Obtain an API Key](#) .
- (Play Integrity only) Project Number
- (Play Integrity only) Service Account Private Key File
- Obtain SHA-256 Certificate Fingerprints. This value is used to validate whether client requests come from the app signing with the expected keystore. To obtain SHA-256 Certificate Fingerprints, use the `keytool` utility provided by Java via the command line:

```
keytool -list -v -alias <key-alias> -keystore <keystore-path>
```

You are prompted to enter the keystore password. From the output, copy the “SHA256” value under “Certificate fingerprints”. It’s a 64-byte hex string. You can add multiple SHA-256 Certificate Fingerprints if needed.



The attestation feature is only available in BTP SDK for Android 5.1 and above. Once attestation is enforced, any requests from an app built via an unsupported SDK, like SAP Mobile Platform SDK, are rejected.

Caution

Please turn on enforcement with caution. If your Mobile Services application is not solely used for the app developed by BTP SDK for Android 5.1 and above, we recommend you deploy a Mobile Services application for enabling Android attestation devices.

Context

With device attestation integrated with Mobile Services, you can safeguard your server API by examining the device's software and hardware environment at runtime, determining whether your server is communicating with a genuine app and device, then deciding whether to accept or reject the client's requests. Administrators can monitor non-compliant Android devices / apps, and enforce compliant runtime checking.

The Android attestation implementation is based on the [Play Integrity API](#) . Previously it was based on the [SafetyNet Attestation API](#) . Google has announced end of support for SafetyNet. In preparation, Play Integrity has been integrated with SAP Mobile Services, and a migration option provided for existing Android apps. SAP recommends that you migrate to the Play Integrity API as soon as possible.

SafetyNet Attestation API Each time the client calls the SafetyNet Attestation API, usage quota is consumed. The default quota per project is 10,000 times per day. You are alerted if you exceed your quota. To monitor

the quota usage, see: [Set Up Quota Monitoring and Alerting](#) . To apply for more quota when needed, see [SafetyNet Attestation API - Quota Request](#) .

Note

When you enable attestation, an event log subscription is created automatically and you are notified through [My Alerts](#) if you exceed quota. Look for a message similar to `Android attestation quota exceeded`.

Play Integrity Attestation API In the same way, each time the client calls the Play Integrity Attestation API, usage quota is consumed. The default quota per project is 10,000 times per day. You are alerted if you exceed your quota. To monitor the quota usage, see: [Set Up Quota Monitoring and Alerting](#) . To apply for more quota when needed, see [Request a Higher Quota](#) (form).

Note

When you enable attestation, an event log subscription is created automatically and you are notified through [My Alerts](#) if you exceed quota. Look for a message similar to `Android attestation quota exceeded`.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#) .
2. Select an application, and [Attestation](#).
3. Select [Pencil icon](#) , and then the [Enabled](#) checkbox.

When you enable attestation, an event log subscription is created automatically, so you are notified through [My Alerts](#) if you exceed quota.

4. In [Edit Android Attestation Settings](#), configure the Android attestation settings, and select [Save](#). Note that for an existing configuration, select [Migrate](#) to migrate to Play Integrity, as described in [Migrating from SecurityNet to Play Integrity](#).

These values are used to validate an Android app's integrity.

Android Attestation Settings

Field	Description
Google API Key (SafetyNet only)	(Required) The access key required to call the SafetyNet Attestation API.
Project Number (Play Integrity only)	(Required) The unique identifier assigned to each Google Cloud Platform (GCP) project. The project number is used in the Android Play Integrity APIs for authentication and managing resources within the project.

Field	Description
Service Account Private Key File (Play Integrity only)	(Required) A service account in Android Play Integrity is a special type of Google account that represents an application, rather than an individual user. It is used to authenticate and authorize the application to access Play Integrity APIs securely, enabling the app to verify its integrity and protect against tampering or unauthorized modifications. You can Upload a new file or Remove a file to replace it.
Package Name	(Required) The name of the application package. The package name is used to check whether the client requests come from the app using the expected package name.
Attestation Token Lifetime	(Required) How long an attestation token is valid, in minutes, hours, or days. When the client passes attestation validation, Mobile Services issues a token. Within the token lifetime, the client doesn't need go through the attestation token flow again. The minimum value of token lifetime is 30 minutes, the maximum value is 7 days.
Debug Token	<p>Once Android attestation is enabled, you can provide a debug token for debugging and testing. This enables you to skip attestation check.</p> <p>To do so, attach the following header when accessing the Mobile Services API:</p> <pre>x-ms-attestation- token:<debug_token_value></pre> <div> <p>Note</p> <p>For security, be cautious about not leaking the debug token value.</p> </div> <p>You can select Generate to generate a new token for debugging, and Revoke to remove the token. The value is masked automatically.</p>
SHA-256 Certificate Fingerprints	(Required) This value is used to validate whether the client requests come from the app signing with the expected keystore. Provide the 64-byte hex string generated by <code>keytool</code> . You can add multiple SHA-256 Certificate Fingerprints, and add or delete values

Once you save, the [Attestation Request Monitor](#) appears, and Android attestation for the selected application begins. Mobile Services validates the attestation token and logs the results for each request, but does not reject requests that do not include an attestation token. The [Status](#) appears as [Unenforced](#).

- When you are ready (keeping in mind the guidelines provided above), select the [Enforce](#) button. Then, only requests that pass attestation validation are accepted, and the rest are rejected. The [Status](#) changes to [Enforced](#).


The [Attestation Request Monitor](#) provides a graph showing the number of requests, in specific categories, by platform and time frame. Change filters to view specific information.

Attention Request Monitor Filters

Filter	Description
Time Frame	Select the time frame from the list, including last hour, last 24 hours, last 7 days, and last month.
Platform	Select the platforms to include, such as Android, iOS or Unknown.

The graph may include these request types:

- [Verified](#): the requests passed the attestation validation.
- [Outdated Client](#): the requests can't provide the attestation token. This may indicate an older application version.
- [Invalid](#): the requests failed the attestation token check.
- [Unknown Origin](#): the requests did not come from an Android device; for example, they may have come from a browser or the Postman platform.

6. (Optional) Under [Android Attestation](#), select  to change settings.
7. Under [SafetyNet Attestation API Quota Monitoring](#) or [Play Integrity Attestation API Quota Monitoring](#) follow the links to learn more about setting up quota monitoring and alerting, and requesting additional quota.

If you exceed quota, you see an event log message similar to `Quota exceedance occurred on <number_of_days> of the last 30 days`. If this happens frequently, consider requesting additional quota.

[Migrating from SecurityNet to Play Integrity \[page 311\]](#)

You can migrate apps from Google SecurityNet to Play Integrity for Android Attestation from the Mobile Services cockpit.

1.5.6.11.1 Migrating from SecurityNet to Play Integrity

You can migrate apps from Google SecurityNet to Play Integrity for Android Attestation from the Mobile Services cockpit.

Prerequisites

- Google Cloud Platform (GCP) project number
- Private key file associated with the service account

Context

Since Google is ending support of SecurityNet for Android Attestation, SAP recommends migrating existing Android apps from SecurityNet to Play Integrity as soon as possible.

Procedure

1. In Mobile Services cockpit, select ► [Mobile Applications](#) ► [Native/MDK](#) ►.
2. Select an application, and [Attestation](#).
3. In [Android Attestation \(SafetyNet, deprecated\)](#), select [Migrate](#).
4. In [Migrate to Play Integrity](#), provide values for these new Play Integrity properties (other SafetyNet values are carried over automatically).
 - Google Cloud Platform (GCP) project number
 - Private key file associated with the service account
5. Review the settings carefully. Once you migrate, SafetyNet settings are replaced by Play Integrity settings and cannot be restored. After migration, the enforcement status is changed to [Unenforced](#).
6. Select [Save](#).

The message `Migration successful` appears once successfully saved.

- The [Android Attestation](#) values change to Play Integrity values, and the [SafetyNet Attestation API Quota Monitoring](#) section is renamed to [Play Integrity Attestation API Quota Monitoring](#).
- In [Attestation Request Monitor](#), Play Integrity values are implemented automatically when you select Android.
- The enforcement status changes to [Unenforced](#). When you are ready, select the [Enforce](#) button to turn enforcement back on, as described in [Configuring Android Attestation](#). Then, only requests that pass attestation validation are accepted, and the rest are rejected. The Status changes to [Enforced](#).

1.5.6.12 Configuring iOS Attestation

(Native/MDK only) Device attestation enables developers and administrators to learn about the software and hardware environment of devices that are trying to connect to enterprise apps and workspaces.

Prerequisites

- Ensure that the correct version of the App Store service is installed on the user's device.
- Know the Bundle ID and Team ID for the application.

The attestation feature is only available in BTP SDK for iOS v9.1 and above. Once attestation is enforced, any requests from an app built via an unsupported SDK, like SAP Mobile Platform SDK, are rejected.


⚠ Caution

Please turn on enforcement with caution. If your Mobile Services application is not solely used for the app developed by BTP SDK for iOS v9.1 and above, we recommend you deploy a separate mobile service application for enabling iOS attestation devices.

Context

With device attestation integrated with Mobile Services, you can safeguard your server API by examining the device's software and hardware environment at runtime, determining whether your server is communicating with a genuine app and device, then deciding whether to accept or reject the client's requests. Administrators can monitor non-compliant iOS devices / apps, and enforce compliant runtime checking.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) > [Native/MDK](#).
2. Select an application, and [Attestation](#).
3. Under [iOS Attestation](#), select , and then the [Enabled](#) checkbox.
4. In [Edit iOS Attestation Settings](#), configure the iOS attestation settings, and select [Save](#).

These values are used to validate an iOS app's integrity.

iOS Attestation Settings

Field	Description
Bundle ID	The bundle ID that is specific to the iOS application you're setting up for iOS attestation, for example, <code>com.sap.attestationSample</code> .
Team ID	The 10-character team ID you use for developing your company's apps. Obtain this value from your developer account.
Environment	The target environment, including Development and Production.
Attestation Token Lifetime	(Required) How long an attestation token is valid, in minutes, hours, or days. When the client passes attestation validation, mobile services issues a token. Within the token lifetime, the client doesn't need go through the attestation token flow again. The minimum value of token lifetime is 30 minutes, the maximum value is 7 days.

Field	Description
Debug Token	<p>Once iOS attestation is enabled, you can provide a debug token for debugging and testing. This enables you to skip attestation check.</p> <p>To do so, attach the following header when accessing the mobile services API:</p> <pre>x-ms-attestation-token:<debug_token_value></pre> <div> <p>Note</p> <p>For security, be cautious about not leaking the debug token value.</p> </div> <p>You can select Generate to generate a new token for debugging, and Revoke to remove the token. The value is masked automatically.</p>

Once you save, the [Attestation Request Monitor](#) appears, and iOS attestation for the selected application begins. Mobile service validates the attestation token and logs the results for each request, but does not reject requests that do not include an attestation token. The [Status](#) appears as [Unenforced](#).

- When you are ready (keeping in mind the guidelines provided above), select the [Enforce](#) button. Then, only requests that pass attestation validation are accepted, and the rest are rejected. The [Status](#) changes to [Enforced](#).

The [Attestation Request Monitor](#) provides a graph showing the number of requests, in specific categories, by platform and time frame. Change filters to view specific information.

Attention Request Monitor Filters

Filter	Description
Time Frame	Select the time frame from the list, including last hour, last 24 hours, last 7 days, and last month.
Platform	Select the platforms to include, such as Android, iOS or Unknown.

The graph may include these request types:

- [Verified](#): the requests passed the attestation validation.
- [Outdated Client](#): the requests can't provide the attestation token. This may indicate an older application version.
- [Invalid](#): the requests failed the attestation token check.
- [Unknown Origin](#): the requests did not come from an iOS device; for example, they may have come from a browser or the Postman platform.

- (Optional) Under [iOS Attestation](#), select  to change settings.

1.5.6.13 Data Protection and Privacy

SAP Mobile Services does not track or store personal data, but tracks data that is related to the mobile services and set-up details.

Context

Data protection is associated with numerous legal requirements and privacy concerns. In addition to compliance with general data privacy regulation, it is necessary to consider compliance with industry-specific legislation in different countries. SAP provides specific features and functions to support compliance with regard to relevant legal requirements, including data protection. SAP does not give any advice on whether these features and functions are the best method to support company, industry, regional, or country-specific requirements. Furthermore, this information should not be taken as advice or a recommendation in regards to additional features that would be required in specific IT environments; decisions related to data protection must be made on a case-by-case basis, taking into consideration the given system landscape and the applicable legal requirements.

Note

SAP does not provide legal advice in any form. SAP software supports data protection compliance by providing security features and specific data protection-relevant functions, such as simplified blocking and deletion of personal data. In many cases, compliance with applicable data protection and privacy laws will not be covered by a product feature. Definitions and other terms used in this document are not taken from a particular legal source.

Important Terms

Term	Definition
Consent	The action of the data subject confirming that the usage of his or her personal data shall be allowed for a given purpose. A consent functionality allows the storage of a consent record in relation to a specific purpose and shows if a data subject has granted, withdrawn, or denied consent.
Deletion	The irreversible destruction of personal data.
Personal data	Any information relating to an identified or identifiable natural person ("data subject"). An identifiable natural person is one who can be identified, directly or indirectly, in particular by reference to an identifier such as a name, an identification number, location data, an online identifier or to one or more factors specific to the physical, physiological, genetic, mental, economic, cultural, or social identity of that natural person.

Term	Definition
Mobile service and set-up	Any information related to using the mobile service and setting up access, authentication, operations, security, and so forth. It may be related to a "data subject" but in the context of processing authentication, data, usage, and so forth.
Purpose	A legal, contractual, or in other form justified reason for the processing of personal data. The assumption is that any purpose has an end that is usually already defined when the purpose starts.

SAP Mobile Services (Cloud Foundry)

Mobile service and set-up details typically include:

- **Tenant Data**

Tenant Data is stored in a SAP BTP mobile services provider account.

When the customer deletes a mobile application, its related data are all purged. If a subaccount\org\space is deleted, mobile applications and related data deployed in them will be eventually purged by our scheduler

- **Application Client Logs, Usage Reports, and Crash Logs**

Client logs, usage data, and crash logs are maintained by SAP Mobile Services, but no sensitive data is stored.

Client application developers can use the SAP SDK for Android and SAP SDK for iOS to build an opt-in form to collect consent for usage analysis from their users. The customer can customize the form for each application. The application user gives consent from the device for specific Usage Report data to be collected. The Usage Report does not contain personal data, only anonymized information about the usage of the app (for example, which pages were opened, how long the pages were open, and so forth). The server records the response, retains the last five versions of the consent form, and the version used to give consent.

Tenant dev/ops personnel set the schedule to purge logs, or can purge them immediately, using the Mobile Services cockpit.

Client logs are kept up to six months unless a custom-defined date is used. To purge client data immediately, please initiate a support ticket. The SAP SDK for Android and SAP SDK for iOS may collect source and location information related to the file in which the logging occurred. Additionally, the app developer may pass a message and its severity.

Client usage data is kept 365 days. To purge usage data immediately, please initiate a support ticket. SAP SDK for Android and SAP SDK for iOS may collect the following usage information (additionally, the app developer can pass event-specific information, such as view name, action name, value, and so forth):

- Platform and version (like "16.6" for iOS)
- Environment (like "Simulator" or "Device")
- Application ID and version
- Device ID and model name

Crash logs are kept for three months. SAP SDK for Android and SAP SDK for iOS may collect the following information for crash logs:

- Platform and version (like “16.6” for iOS)
- Platform architecture (like “Simulator” or “architecture”)
- Stack trace and Thread pool
- Application ID and version
- Build and SDK version
- Device model name and version
- Device locale and screen size
- Record timestamp
- Network connectivity
- **Network Trace**
Mobile Services does not deal with personal data directly but rather acts as a proxy to the back end. The platform has the option to temporarily capture network traffic to the back end for debugging purposes. This network trace information might contain personal data exchanged from device to back end. Enabling/disabling network trace is audited, and accessing network trace (downloading from Mobile Services cockpit) is also audited in our audit log table. Network trace is retained for 7 days. We also provide functionality to purge network trace manually.
- **Export User Data**
Mobile Services provide feature in cockpit to export user data based on username in each application. It can be used to export all data relevant to a user within the application.
- **Applications that you Create**
The cloud service does not store personal data, but only synchronizes data exchanged between the mobile device and the back end.
Developers can create applications that store sensitive data in an offline data store. In this case, developers should set the client password policy that is used to unlock the data store when an application initializes, as described in *Defining Client Password Policy*, and should also encrypt the contents of the offline data store using the `storeEncryptionKey` method. This ensures that data is automatically encrypted when the store is used for the first time.
WeChat Micro App applications collect users' nickname and profile photo, only after users provide consent and their WeChat ID. The purpose is to help users log on to Mobile Services cockpit using Single Sign On from WeChat (client app on devices), and to help Micro App applications access the nickname and profile photo for personalization.
After users unsubscribe, the collected information is deleted. When, customers / tenants cancel mobile services or delete WeChat Micro App, the collected information is deleted.

Related Information

[Defining Client Passcode Policy \[page 62\]](#)

1.5.7 Troubleshooting: Common Issues

Overview of common issues.

Note

You can also check [Guided Answers](#) for interactive documentation designed to help troubleshoot issues, navigate processes and guide through tasks. For example, search for Mobile Services to see its [Guided Answers Tree](#).

Problems Uploading Files in App Lab using Internet Explorer

Problem

You upload a file and encounter an error like `Upload Failure Failed to upload application version <xxxxxxx.xxx> with error:0, message: Unknown failure to API service`. This can happen if you are uploading application files that contain Unicode characters, using an Internet Explorer browser.

The issue stems from Internet Explorer's method of encoding the filename as a request parameter. With the `app-info-service`, no encoding is necessary because the filename is in the body of the request. However, for the `app-binary-service`, the specified app name is in the request parameters of the `app-binary-service` upload API endpoint. As a request parameter, the app name is encoded differently on Internet Explorer than Chrome/Firefox.

Workaround

If you are uploading application files that contain Unicode characters, use a Firefox or Chrome browser instead of an Internet Explorer browser.

Alternatively, you could rename the application file that contains Unicode characters so that it includes the filename.

Disappearing Scroll Bar Issue in Internet Explorer

Problem

After using the browser's zoom-in feature, the scroll bar appears at first, but then disappears after a short time. For example, when viewing the usage reports for a selected application, and using the zoom-in feature, the scroll bar first appears at the bottom of the usage window, but then disappears.

Workaround

This is a known issue in Internet Explorer (in a Chrome browser, the scroll bar persists after zooming in). As a workaround, system administrators can use the Microsoft Edge browser available with the Windows 10 operating system.

Error 403 Accessing an Existing Application

Problem

You access an application in the cockpit using an existing registration, and encounter the error message:

HTTP Status 403 - The registered user for application connection does not Match the user.

This can happen if you update the authentication type associated with the application configuration, and then access the application using an existing registration.

Workaround

The authenticated user name used to access the application must match the authenticated user name that was established during registration. Changing the authentication type can result in a different user name being established after successful authentication. To correct the problem:

1. Delete the existing registration via the cockpit.
2. The client should re-register.

Cannot Establish OData Connectivity from Mobile Services

Problem

When attempting to test a connection from Mobile Services cockpit to an OData back end through SAP Cloud Connector, the following message appears:

Check the path provided and destination configuration, ensure that the OData service is available.

Workaround

This typically occurs if the Trust relationship is not enabled between Mobile Services and Cloud Connector. The Mobile Service java applications (mobilejava, mobilepreview) must be marked as Trusted in On-Premise Cloud Connector configuration. See [Configure Trust](#).

Cannot View Event Logs from the Home Screen

Problem

You cannot view event logs from the Home screen on Internet Explorer 11.

Workaround

You may need to configure Internet Explorer 11 to view the XML event log files:

1. Go to ► [Internet Options](#) ► [Content](#) ► [Feeds and Web Slices](#) ► [Settings](#) ►.
2. Uncheck [Turn on feed reading view](#).

Offline Applications Cannot Connect to Back Ends

Problem

If the back-end server does not support ClientCertificate authentication, offline applications cannot connect.

Workaround

For offline applications, do not specify the keystore location.

Offline Store Reports Configuration Not Saved

Problem

In the ► [Mobile Offline Access](#) ► [Offline Store Upload Policy](#) ► tab, when attempting to [Enable Offline Store Upload](#) for the selected application, Mobile Services cockpit reports [Configuration not saved](#).

This can indicate that a Cloud Foundry trial has expired, or will expire during the specified time frame. To verify, check the event log for an entry like:

```
failed to update feature Mobile Offline Access for application com.<app_name>,
error: Validate business configuration of service
Mobile Offline Access failed: 400 Bad Request, Upload policy expired time
exceeds the limit of trial plan.
```

Solution

The value entered for [Delete Offline Store After](#) should not exceed 3 days for the upload policy of a trial account. If it does, please enter a lower value.

Incorrect Password Causes Web Application Registration Problems

Problem

If you enter an incorrect password during client web application registration, and then enter the correct password in the pop-up challenge dialog, SAP Mobile Services denies application registration.

Workaround

1. Click [Cancel](#) in the challenge dialog.
2. Click [OK](#) when you see the error: `Username or password incorrect`.
3. On the registration page, enter the correct user name and password again. The web application registers, and then opens.

Slow Performance when Downloading Large Logs

Problem

When downloading large log files from the cockpit, performance is very slow.

Solution

To improve performance, a user with the Administrator/Help Desk role can use the HTTP API from the command line:

```
/Admin/log?from=yyyyMMddHHmmss&to=yyyyMMddHHmmss
```

Where either the "from" or "to" parameter is optional. No additional header is required for the API.

The output contains the server log, client log and application trace, in plain text format.

Not Authorized to Access SAP Mobile Services

Problem

When attempting to log in to SAP Mobile Services, you receive the message *You are not authorized to access SAP BTP mobile services. Log in with the correct role, or contact the System Administrator.*

Solution

The user log on account must contain the correct roles before SAP Mobile Services grants access to users. See *Set Up Customer Accounts* for information about roles. Ask the account administrator to verify that the account includes an appropriate role.

Error 403 Accessing Mobile Services cockpit

Problem

If a custom identity provider was configured in SAP BTP cockpit, you may get a 403 error when you access Mobile Services cockpit.

Workaround

1. Verify that the appropriate roles have been granted to the correct user ID. See *Set Up Customer Accounts*.

Note

The user ID in a role assignment must match the authenticated subject in the SAML Assertion that the IdP returns to SAP BTP.

2. To troubleshoot the error, install a browser plugin that helps you capture the SAML Request/Response, for example, the SAMLTracer plugin in Firefox or SAML Chrome Panel.
3. Enable and start the trace, and access the Mobile Services cockpit URL.
4. Examine the SAMLAssertion that the IdP returns to SAP BTP, specifically the `User ID Source` in the IdP configuration. See [ID Federation with the Corporate Identity Provider](#).

5. By default, the value of `NameID` is used as the authenticated user ID. Verify that this value matches the user ID in the role assignment.

SAP Mobile Cards: Query URL Card Can't be Downloaded

Problem

Users encounter a situation where they cannot download migrated SAP Mobile Cards on the client side.

Workaround

This may indicate that the landscape was upgraded to a higher version. To correct the problem:

- (Development) If the card is a development card, the user should open it and save it again in Mobile Services cockpit. Then, download the card on the client side.
- (Production) If the card is a productive card, the user should copy a development card, save it again, and republish it as a productive card. Then, download the card on the client side.

MDK App Fails to Launch with Error

Problem

From an Android device, you attempt to open a Mobile Development Kit (MDK) app using an URL. Instead, the app does not open, and the message `Can not connect to the server, please try again later` appears.

Workaround

This may indicate that the scheme was not entered, or was not entered correctly, in the Cloud Build configuration page when the app was generated. From Mobile Services cockpit, select the application and check its Cloud Build configuration. Be sure the scheme is entered correctly in the [URL Scheme](#) field. The [URL Scheme](#) is used for both iOS and Android builds.

In this example, you enter `sapmobilesvcs` as the scheme in [URL Scheme](#), but not the full Onboarding URL:

```
sapmobilesvcs: //?AppId=com.sap.mk.mdk&ClientId=xxxxxx-xxxxxx-xxxx-xxxx-xxxxxxx&SapCloudPlatformEndpoint=https://hcpms-xxxxxx.hanatrial.ondemand.com&AuthorizationEndpointUrl=https://oauthservices-xxxxxx.hanatrial.ondemand.com/oauth2/api/v1/authorize&RedirectUrl=https://oauthservices-xxxxxx.hanatrial.ondemand.com&TokenUrl=https://oauthservices-xxxxxx.hanatrial.ondemand.com/oauth2/api/v1/token
```

See:

- [Configuring and Building Apps with the Cloud Build Service](#) (topic group)
- [Supported Client Types, Packaging Details, and Build Options](#) (reference)

Unable to Access Mobile Services from the BTP Trial Account

Problem

When setting up a customer account in SAP BTP cockpit, the Mobile Services option does not appear from the [Support](#) link.

Solution

This may indicate that the Cloud Foundry landscape is an extended landscape, since extended landscapes do not appear in the [Support](#) link. To use Mobile Services cockpit in an extended Cloud Foundry landscape, you must create a mobile services instance in SAP BTP cockpit. To access the Mobile Services cockpit, select the mobile services instance link, [Open Dashboard](#). For more information see *Set Up Customer Accounts* and *Using Mobile Services Cockpit in an Extended Landscape*.

Related Information

[Set Up Customer Accounts \[page 20\]](#)

[Configuring and Building Apps with the Cloud Build Service \[page 157\]](#)

[Supported Build Types, Client Types, Packaging Details, and Build Options \[page 161\]](#)

[Set Up Customer Accounts \[page 20\]](#)

1.6 Development

Applications can be developed using REST API or OData SDK.

To develop applications using OData SDK, see the latest version of [Application Development in SAP Mobile Platform](#).

For using SAP BTP SDK, see:

- [SAP SDK for Android](#)
- [SAP SDK for iOS](#)

1.6.1 REST API Application Development Overview

The REST Services, distributed as part of the SAP Mobile Services, enables standard HTTP client applications running in any platform to leverage mobile platform for security and push features.

Build client applications using third-party developer tools (JavaScript framework and helper libraries), native client libraries, or the libraries provided by the platform OData SDK (iOS and Android platforms only).

The mobile platform enables you to manage and monitor the applications, and provides support for native push notification: Apple Push Notification service (APNs), BlackBerry Internet/Enterprise Service (BIS/BES),

Firebase Cloud Messaging (FCM), Windows Notification Service (WNS), or Microsoft Push Notification Service (MPNS).

Application developers must first register an application connection using the REST client and provide the device information, such as device type, password capability, and so on. Once registered, an application can retrieve and update the application connection settings through the REST API. You can enable or disable the push notification only after registering.

Note

You can delete an application connection using the REST API, as long as the application is not in use. Any data that is stored in the custom `string` of the application connection properties is lost.

During initialization, a client application can download resources (metadata files, multimedia files, and so on), using the resource bundles service. After downloading resources, the application can access OData-compatible data sources through the proxy service, and receive native push notifications triggered by SAP Gateway if push properties are configured and enabled.

This development approach supports:

- Registration (creating an application connection)
- Authentication
- Native push notification
- Usage reporting

Cloud solutions do not have a Product Availability Matrix (PAM). For more information about cloud solution product versions contact an SAP representative.

Related Information

[Set Up Customer Accounts \[page 20\]](#)

[Push Overview \[page 36\]](#)

[Custom Push \[page 147\]](#)

1.6.1.1 Configure Applications in Mobile Services Cockpit

Configure an application definition that enables you to manage and monitor the applications using Mobile Services cockpit.

[Defining Applications \(Cloud Foundry\) \[page 325\]](#)

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

[Defining Connectivity \[page 330\]](#)

Define destinations for the selected application. You can also edit Mobile and On-Premise destinations.

[Configuring App Security \[page 332\]](#)

Configure security at the application level for the Cloud Foundry environment.

[Defining Push Notifications \[page 342\]](#)

Configure push-related settings for the selected application.

[Uploading Client Resources \[page 345\]](#)

Upload client resources (or resource bundles) and manage service keys for the selected application.

1.6.1.1.1 Defining Applications (Cloud Foundry)

Create a new application definition, which enables you to use the Mobile Services cockpit to manage the application.

Context

When you define an application, choose a configuration template. Keep in mind:

- When you select a Native, Hybrid, or Mobile Development Kit template, the most typical features for the application type are selected automatically. You can add or remove features at this time, or take care of it later.
- Some features, such as Mobile Settings Exchange, are required, so you cannot remove them.
- Other features have dependencies, so you either cannot remove them, or must remove them in a particular order. For example, Mobile Offline Access requires Connectivity, and Mobile Cloud Build requires Mobile App Catalog.

After you define the application, configure the assigned features, or add more features. See *Managing Application Features* and *Configuring Assigned Features*.

Procedure

1. In Mobile Services cockpit, select  [Mobile Applications](#)  [Native/MDK](#) , and click [New](#).

Note that some service plans impose a limit on the number of applications that you can implement. When you reach the limit, a message appears, such as "The current activity cannot be performed due to limitations on the current plan". To mitigate, delete one of the applications. See *Service Plans* for information about service plan limits.

2. In [New Application](#), on the [Basic Info](#) page, enter the application properties, followed by [Next](#):

Application Properties

Field	Value
ID	<p>Unique identifier for the application, in reverse-domain notation. This is the application or bundled identifier that is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Mobile Services, and client applications use the application ID when sending requests to the server.</p> <p>An application ID:</p> <ul style="list-style-type: none"> • Must be unique • Must start with an alphabetic character • Can contain only alphanumeric characters, underscores, and periods • Can contain up to 64 characters • Cannot include spaces • Cannot begin with a period, and cannot contain two consecutive periods <p>We recommend that you assign IDs that contain a minimum of two periods, for example, <code>com.sap.mobile.app1</code>.</p> <p>If you are building an Android application, its ID must follow the Google defined rules, or the Android build fails:</p> <ul style="list-style-type: none"> • The ID must have at least two segments (one or more periods). • Each segment must start with a letter. • All characters must be alphanumeric or an underscore [a-zA-Z0-9_].
Name	The application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.
Description	(Optional) The description can contain up to 255 alphanumeric and special characters.
Vendor	(Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long.
License Type	The license type that you want to use for your account: "resources" (default), "build-code", "free", "standard (Users)", or "b2c (Consumer Edition)". The "kernel-service" plan is used for an application that is managed in a single-app cockpit (Software as a Service, SaaS). The "lite" plan is available only for trial subaccounts. The license determines capabilities and charges. See <i>Service Plans</i> and <i>Changing Service Plans at the App Level</i> for additional information.
Domain of Application Route	<p>In regions that support multiple domains, select the application route through which end users can reach the application. Routes are associated with a space, and are configured in SAP BTP cockpit. The drop-down list shows all shared and custom domains that are available for the customer organization.</p> <div> <p>Note</p> <p>If you want to use a custom domain, you can create one according to information in What Is Custom Domain.</p> </div>

Field	Value
Client Connection Timeout	<p>(Optional) The maximum time in milliseconds before a client connection times out in your environment. After that timeout period, the connection is closed. If set to 0, the timeout is disabled. Default is 120000 milliseconds.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Session Timeout	<p>(Optional) The number of minutes a session can remain idle before the server terminates it automatically. Default value is 15 minutes.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Enable Compression	<p>(Optional) Compresses resources before sending them to the client. By default text resources above 1KB is compressed.</p> <p>Note No longer appears after upgrading to mobile application service.</p>
Zero Maintenance Downtime	<p>(Optional) Indicates whether zero maintenance downtime is enabled.</p> <p>Note No longer appears after upgrading to mobile application service.</p>

- On the [Security Settings](#) page, enter configuration values and then select [Next](#).

For environments where Identity Authentication service (IAS) has been configured, you have the option to select either the IAS (Identity Authentication service) or XSUAA (extended UAA) security configuration when you create a new mobile application. See *Configuring IAS Security* for information about configuring Identity Authentication service.

- [XSUAA Settings](#) - select this option to use the SAP Authorization and Trust Management service method for authentication.
- [IAS Settings](#) - select this option to use the SAP Identity Authentication service method for authentication.

Note

You cannot switch a mobile application between XSUAA and IAS, but you can switch between the default and existing instances.

See the following tables for configuration values, depending on your selection. You can modify these values later on the [Security](#) tab, as described in *Configuring App Security*.

XSUAA Settings

Field	Value
XSUAA Service	<p>Select the XSUAA authentication and authorization service to use. You can select Default Instance to create a new instance, or you can select an existing service from the list.</p> <p>Using an existing XSUAA instance is useful if you already have a Cloud Foundry application deployed and want to connect from Mobile Services to this application. In this case it can be handy to re-use an existing XSUAA instance.</p> <p>Select the default instance for other scenarios.</p>
xs-security.json	<p>A JSON file that defines the authentication methods and authorization types used to access your application. See Application Security Descriptor Configuration Syntax for additional information.</p>
Token Lifetime	<p>Enter the token lifetime, and select the units (Hours or Minutes). The default token lifetime is 15 minutes, and the allowed range is 10 minutes to 24 hours.</p> <p>See Application Security Descriptor Configuration Syntax for additional information.</p>
Refresh Token Lifetime	<p>Enter the refresh token lifetime, and select the units (Days, Hours, or Minutes). The value must be greater than the Token Lifetime value.</p> <p>See Application Security Descriptor Configuration Syntax for additional information.</p>
Approved Providers	<p>Indicate whether the app should support all approved providers or only selected providers.</p>
System Attributes in Token	<p>Indicate groups, role collections or both.</p>


IAS Settings

Field	Value
IAS Service	<p>Select the IAS authentication and authorization service to use. You can select Default Instance to create a new instance, or you can select an existing service from the list.</p>
Security-config.json	<p>A JSON file that defines the authentication methods and authorization types used to access your application.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>

Field	Value
Token Lifetime	<p>Enter the token lifetime, and select the units (Hours or Minutes). The default token lifetime is 15 minutes, and the allowed range is 10 minutes to one hour.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>
Refresh Token Lifetime	<p>Enter the refresh token lifetime, and select the units (Days, Hours, or Minutes). The default refresh token lifetime is 30 days, and the allowed range is 1 hour to 180 days.</p> <p>See Reference Information for the Identity Service of SAP BTP for additional information.</p>

- On the [Role Settings](#) page, enter roles and then select [Next](#).

The [Enable Role Settings](#) option is selected by default. When selected you must add one or more roles to the list. Only users who are assigned these roles can access the app, which helps ensure better security. SAP recommends enabling roles.

Enter one or more role names, separated by either commas or the [Return/Enter](#) key, for example "Developer, Sales, Manager". To learn more about default Cloud Foundry roles, see [Orgs, Spaces, Roles, and Permissions](#) .

Once the roles have been entered, you can maintain them through the [Security](#) tab for the app as described in [Configuring App Security](#).

- On the [Assign Features](#) page, select one of the configuration templates.

- Native – applications built using the native SDKs for iOS or Android.
- Mobile Development Kit - metadata-based applications.
- Hybrid – Kapsel container-based applications.

The features selected for the template appear. You can select or remove features (except for required features) at this time. Later you can make further changes as described in [Managing Application Features](#).

Basic Features

Feature	Description
Mobile App Catalog	Upload mobile application artifacts for beta testing, and for deployment to external services.
Mobile App Update	Upload new versions of a hybrid or Mobile Development Kit application.
Mobile Augmented Reality	Allows you to manage client augmented reality resources that can be accessed from mobile applications.
Mobile Settings Exchange	Handles device registrations and provides exchange of general settings between mobile client and server, such as client policies.

Feature	Description
Mobile Client Resources	Add client resources to an application.
Mobile Connectivity	Configure routes to back ends.
Mobile Offline Access	Enable secure, offline access to data on the device.
Mobile Push Notification	Register devices to receive native push notifications.
Mobile Client Log Upload	Enable application to upload application log files and to analyze them on the server.
Mobile Network Trace	Enable application to trace network activity based on user name or content type.
Mobile Sample OData ESPM	Use OData sample service during development and testing.
Mobile Client Usage and User Feedback	Enable application to upload client usage and feedback information and analyze it on the server.
Mobile Cloud Build	Build binaries for your standard and customized Mobile Development Kit (MDK) and SAP Asset Manager clients, and enable them to use the SAP Mobile Platform SDK.

6. Select *Finish* and confirm to create the application definition. The Info tab appears with current settings.

1.6.1.1.2 Defining Connectivity

Define destinations for the selected application. You can also edit Mobile and On-Premise destinations.

Context

The Mobile Connectivity feature of SAP Mobile Services allows you to define the connectivity to back-end systems that the application can use. You can define any number of destinations to different back-ends. Those destinations are to be used exclusively by the application for which they are configured. You can restrict access to allowed paths. For applications that access Web services containing relative URLs, you can add the relative paths to enable the product to handle requests correctly. You can implement service keys for authentication.

In Mobile Services cockpit, you can view the properties of Fiori applications and connections that were developed using other tools and imported into SAP Mobile Services, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#) [Native/MDK](#) or [SAP Mobile Cards](#).

If you select Native/MDK app, you will view a list of mobile applications, with columns for Application ID, Name, Vendor, License Type, State, Outdated, and Creation Date.

If you select Mobile Cards, you will view a list of mobile card templates, with columns Name, Destination/SAP Client/Site ID, Status, Version, Card Template Class, Card Template Type, and Actions.

2. For Native/MDK apps, select an application, then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first).

For Mobile Cards, select [Features](#), then select [Mobile Connectivity](#) under [Assigned Features](#) (or add it first).

3. For [Configuration](#) under [Mobile Destinations](#), you can view current destinations for the selected application.

Destinations

Field	Value
Name	The destination name.
URL	The destination URL.
Rewrite Mode	For application back-end connections, the rewrite mode defines how the mobile services handles request and response messages. To enable applications that use external back ends to run offline, select one of the supported rewrite modes.
SSO Mechanism/Authentication	The single sign-on or authentication security methods employed for the destination.
Actions	<p>The actions available, such as edit or delete a connection, ping a back-end connection, and test an OData application destination.</p> <p>If an action is not supported, the icon is grayed out or absent. For example, pinging and testing OData destinations are not supported for some SSO methods. Use the Launch in Browser icon to test connectivity using the mobile application URL in a separate web browser.</p>

4. (Optional) Select [+](#), and use the [Create Destination](#) dialog to create a new destination. See [Creating a Destination](#) for details.
5. Select a row to view its settings in the Destination Overview.

The overview varies by configuration, but common sections include:

- Info – basic configuration settings.
- Rewrite Method – rewrite URL settings.
- Security – important security settings.
- Custom Headers – key:value pairs defined for static headers, or a cookie value with a variable.

Note

You can select [Edit](#) to edit some settings, or [Ping](#) to test the connection if Ping is supported for the SSO method used.

6. (Optional) From the [Service Keys](#) tab, for some features you can implement a service key, which enables an application to access a service instance using a service key as its credentials. The feature must be able to support service keys. If you do not see the [Service Keys](#) tab for a feature, the option is not available. See [Service Keys](#).
7. (Optional) Select the [Info](#) tab to see useful URLs.

1.6.1.1.3 Configuring App Security

Configure security at the application level for the Cloud Foundry environment.

Context

When you define a new application in the Cloud Foundry environment, an OAuth client is created automatically by the server. A non-null default value is set for client ID and the redirect URL once the application is created.

If you select [OAuth](#) or [SAML](#) for app security authentication, API Key is not enabled. If API Key is configured as part of SDK bootstrapping, services such as setting exchange, log upload, and usage upload are accessed before Mobile Services authentication.

If you select [API Key Only](#) for app security authentication, and enable API Key, anonymous access is used. This is similar to the "No auth" option that is available in the Neo landscape. This combination also provides access to client resources. If you select [API Key Only](#), you must enable API Key. The `x-smp-deviceid` header is mandatory when using API Key. You can use an arbitrary value for the device ID when testing using Postman.

To increase application security, you can provide a specific list or a range of [Allowed IP/CIDR](#) addresses and notations. Requests for the allowed list are performed; requests for addresses or notations that are not specified are ignored. If no entries are provided, there is no restriction, and requests are performed for all addresses.

Security Recommendations

Keep in mind these recommendations when configuring app security:

- Apps should enable the passcode policy.
- Apps should primarily attempt to use the OAuth security configuration.
- Service Keys should favor the X.509 type.
- Review the Audit Logs regularly.
- For critical iOS and Android applications, consider implementing the attestation feature for enhanced security.
- Certificates should be renewed regularly and before they expire to avoid any expiration issues.

For more details see [SAP BTP Security Recommendations](#) and filter on the Mobile Services component.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, and **Security**. View the current security settings, or select the edit button to make changes to a section.
3. (Optional) Lock the application so others can't make changes to the same application.
4. Under **Application Settings**, view or edit settings.
 1. (Web applications only) Select **CSRF Protection**
 2. For **Security Configuration**, select a security option:
 - **API Key Only** – uses an API key for authentication. The key is generated or you can select additional keys in a later step. No other authentication method can be used with this choice.
 - **Basic** – enables stand-alone basic authentication either with or without Cloud Connector. You must download and import a Trust Configuration into your cloud sub-account as described in *Configuring Security Trust*.
 - (Does not appear if IAS is selected) **Basic with SAP ID Service** – authentication delegates to the SAP ID service. If your SAP BTP account uses a different identity provider, this choice fails.
 - **OAuth** – an open protocol for securely authorizing applications using a standard method. SAP BTP must be the authorization server. This is the default option, and the OAuth client is generated automatically. You can use API keys with this choice.
The Proof Key for Code Exchange (PKCE), by OAuth Public Clients, is supported for Authorization Code Grant when using OAuth authentication type. The app must implement the OAuth APIs used by the SDK.
 - **SAML** – uses SAML 2.0 supplied by SAP BTP (the SAP ID service authenticates users against both SAP user accounts and SCN accounts). You can use API keys with this choice.
 - **HTTP** – enables you to use an HTTP URL to access OAuth endpoints by passing a valid bearer token to the back end for authentication. This is similar to the HTTP module available in SAP Mobile Platform, and is especially useful when migrating applications from SAP Mobile Platform to Mobile Services. You must download and import a Trust Configuration into your cloud sub-account as described in *Configuring Security Trust*.
 3. If you select **OAuth** as the security configuration (the default), select the edit button to configure application settings. Click **OK** to save changes.

In **Edit Application Settings**, under **OAuth Clients**, click the **+** icon. You can add or remove OAuth clients or regenerate redirect URLs.

OAuth Client Settings

Field	Description
Client ID	To regenerate the client ID, select an ID and then Regenerate . The ID is automatically generated, and identifies the application client to the authorization server.
Redirect URL	The redirect URL for the OAuth client.

Under **OAuth APIs** you can view and copy the URLs for **OAuth Authorization** and **OAuth Token**.

Under [Cross Context SSO](#), you can enable identity transfer for SSO. See [Configuring Cross Context SSO](#) for information about setting up identity transfer for use in application onboarding.

Cross Context SSO Settings

Field	Description
Enable Cross Context SSO	Select this option to enable cross context SSO. Additional fields appear.
SSO Token Timeout	How long before the temporary login token expires in seconds. The default is 30 seconds.
SSO Onboarding URL	(Read/copy only) The onboarding URL.
SSO Onboarding URL for Android Widget Support (Does not appear if IAS is selected)	(Read/copy only) The onboarding URL that supports the Android Widget when it is enabled. You can make this link available to users.


Note

IAS-based apps do not support read-only tokens. When you configure security for IAS-based apps and enable Cross Context SSO, the "SSO Onboarding URL for Android Widget Support" option does not appear.

Under [QR Code Signature Settings](#) you can enable use of digital QR code signatures for onboarding. See [Configuring Digitally-Signed QR Codes](#) for information about setting up QR code signature settings for onboarding users.

QR Code Signature Settings

Field	Description
Public Key	The public key is used on the client side to verify the digital signature.
Private Key	The private key is used on the server side to sign the QR Code, and is not exposed to any client.

(Mobile Development Kit only) Once the Public Key appears, you can select  to download a file containing the key. This is useful when working with a developer or to keep a back-up copy.

(Mobile Development Kit only) You can also select  to copy the public key to the clipboard, and then paste the contents elsewhere.

- If you select [Basic with SAP ID Service](#), [Basic](#), or [SAML](#) instead of [OAuth](#), you are notified that the change affects the application availability for the device user, and that an application restart is required to update the application's configuration. Select [OK](#).

If you select [Basic](#), provide the [Basic Settings](#), and select [OK](#). Also, the [Mobile Connectivity](#) feature is assigned automatically to the application to provide a proxy service, and cannot be removed (navigate to the [Info](#) tab to verify that [Mobile Connectivity](#) appears under [Assigned Features](#)).

Basic Settings

Field	Description
URL	The URL (endpoint) to use for basic authentication. Mobile Services uses the URL to authenticate mobile apps by sending an incorrect test user name/password to the customer-configured endpoint. The endpoint must return a 401 (Unauthorized) response message to demonstrate that it can reject bad credentials that are sent using <code>Authorization: Basic</code> header. This test is done according to RFC 2617 (as described in: https://tools.ietf.org/html/rfc2617).
Use Cloud Connector	Enables you to use cloud connector.
Cloud Connector Location ID	Your cloud connector identifier.

- Once you select [HTTP](#), if you check [SAP Destination service](#), you must create a destination in SAP BTP Cockpit first, and enter the Cloud Destination that you just created. Or, you can enter an [URL](#) directly. Then input the [Response Header for User Name Extraction](#) fields, and select [OK](#) to save and confirm. The client needs to attach a valid bearer token in the header to access Mobile Service endpoints. The bearer token may be acquired from another Identity Provider. Mobile Services then sends the bearer token to the configured Cloud Destination or URL for validation. Once the token is validated using the "URL" provided, Mobile Services uses the [Response Header for User Name Extraction](#) field value as the header name to retrieve the user name from the response.

HTTP Settings

Field	Description
SAP Destination service	Select to use a cloud destination to validate the given bearer token. You must provide the Cloud Destination Name.
Cloud Destination Name	You must first configure a cloud destination in the SAP BTP cockpit. Access your subaccount, and then navigate to ► Connectivity ► Destinations ► to create a destination, as described in Managing Destinations . The destination's URL should also meet the requirement described in the URL field. Then input the destination name to this field.
Response Header for User Name Extraction	The header name that Mobile Services uses to extract the user name from the URL response, when it passes the bearer token validation.
URL	Enter the URL to use to validate the given bearer token. This URL must support the GET method, and be able to extract the "Authorization" header as the bearer token for verification. If the token passes validation, this URL must respond with 200 HTTP status, while attaching the user's name to the response header. If the token fails validation, this URL must respond with 401 HTTP status.
Use Cloud Connector	Enable to use a cloud connector.
Cloud Connector Location ID	The cloud connector identifier.

Field	Description
Enable User Attributes	When enabled, the app can access additional attributes for the current user, other than userID, such as email address, groups, addresses, and so forth. The attributes must be in a specific format. For more information, see <i>Enabling User Attributes</i> in the paragraph that follows.

Enabling User Attributes

When you authenticate with an Identity provider, it might provide additional attributes for the current logged-in user other than userID, like email, groups, address etc. For SAML 2.0, the attributes are included in the assertion as one attribute statement with multiple values in it. For HTTP authentication, Mobile Services allows HTTP endpoints to provide user attributes via this option. If you select [Enable User Attributes](#), then Mobile Services parsed the URL or destination's response body in JSON format, and propagate these attributes to the SAML attributes while generating the Mobile Service internal SAML assertion.

Here is an example of the response:

```
{
  "email" : "someone@sap.com",
  "group": "[ 'Group1', 'Group2' ]",
  "department": "IT"
}
```

With this function, you can map different user attributes to different roles to achieve the access control. The configuration step as following:

1. In Mobile Services cockpit, create a role. To do so, navigate to [Mobile Application](#) [Native/MDK](#), select the application you are using, then click the [Security](#) tab. In the [Role Settings](#) section, click the [Edit](#) button, check [Enable Role Settings](#), enter a role name, and then save it.
2. In SAP BTP cockpit, create a role collection. To do so, go to your sub-account, navigate to [Security](#) [Role Collection](#), click the [Create](#) button, give the role collection a name, and then click the [Edit](#) button of this role collection to edit the following information:
 - [Roles](#): select the Role Name and Template Name with the Role you created in the first step, and select the current app's Application Identifier.
 - [Attribute Mappings](#): select the Identity provider for Mobile Services (see [Configuring Security Trust](#) for information). Then enter the "Attribute" and "Value" according to the URL or Destination's response body's key and value. If current user's attribute and value matches with the role collection's definition, the corresponding role will be assigned to the user while login.
5. Under [XSUAA Settings](#), view or edit extended services for UAA (XSUAA). See [Demystifying XSUAA in SAP Cloud Foundry](#) for additional information.

If the [XSUAA Settings](#) section does not appear, either the feature is not supported for your configuration or the [IAS Settings](#) option was selected instead. If the edit button does not appear, then values are read-only. When this option is selected, all services, such as registration, are protected with CSRF tokens. Proxied endpoints aren't protected, since they are likely to be protected on the back end..

Note

You cannot migrate existing mobile apps that are configured with XSUAA to IAS. IAS is only supported for new mobile apps. You must recreate the mobile app and configure the new app with IAS settings.

XSUAA Settings

Field	Description
XSUAA Service	<p>Select Default Instance or an existing service instance from the list.</p> <p>Using an existing XSUAA service instance is useful if you already have a Cloud Foundry application deployed and want to connect to this application from Mobile Services.</p> <p>Select the default instance for other scenarios. Note that if you select the default, you can select OAuth Settings.</p>
xs-security.json	<p>Select the JSON file that defines the authentication methods and authorization types used to access your application. See Application Security Descriptor Configuration Syntax for additional information.</p>
Token Lifetime	<p>Enter the lifetime for a token in number of days, hours, or minutes (12 hours is the default). The entry is subject to any limitations imposed by the server. The maximum token lifetime is 86400 seconds - 24 hours.</p> <p>If user credentials are changed or the user account is disabled, the OAuth tokens are automatically revoked, and the user is not able to access mobile application content.</p>
Refresh Token Lifetime	<p>Enter the maximum lifetime for a refresh token in number of days, hours, or minutes (7 days is the default). The entry is subject to any limitations imposed by the server. The maximum refresh token lifetime is 31536000 seconds - 365 days.</p>

Field	Description
Approved Providers	<p>Indicate the approved identity providers for the application:</p> <ul style="list-style-type: none"> • All (default) – all identity providers are approved for the app. • Selected Providers – enter one or more approved identity providers for the app (hit <i>Enter</i> after each entry). <div data-bbox="890 589 1402 808"> <p>Note</p> <p>Since Mobile Services cockpit cannot validate the identity provider entries, it is very important to enter them correctly. Otherwise, users will not be able to log in successfully.</p> </div> <p>The Selected Providers option enables you to be specific about the identity providers for the app, especially if you have multiple providers configured for the subaccount.</p> <p>For example, say you have two identity providers configured for the subaccount – "sap.default" (origin of default identity provider "SAP ID Service") and "hanamobile_sci". Assume you have two mobile applications, and one application needs to be protected by the "SAP ID Service", and another one by "hanamobile_sci". You can configure "sap.default" as the approved provider for the first application, and "hanamobile_sci" for second application.</p>

Field	Description
System Attributes in Token	<p>The <code>system-attributes</code> property controls which properties should be included in the JWT token generated by an xsuaa service instance. Its purpose is to decrease the size of JWT by excluding some properties, especially when many groups have been assigned to a user in IDP.</p> <p>You can configure <code>system-attributes</code> for a mobile applications xsuaa service instance using:</p> <ul style="list-style-type: none"> • <code>Rolecollections</code> (default for new applications) • <code>Groups</code> <p>For legacy applications (those that were created before the introduction of this feature, and do not have this property set), the value of <code>system-attributes</code> might be:</p> <ul style="list-style-type: none"> • If <code>system-attributes</code> was set in <code>xs-security.json</code>, retrieve the property value and use it as the <code>system-attributes</code> value. • If not, use the <code>Groups</code> or <code>Rolecollections</code> value as the <code>system-attributes</code> value. <p>This option is only available for OAuth. To learn more, see Application Security Descriptor Configuration Syntax.</p>


- Under [IAS Settings](#), view or edit SAP Identity Authentication service settings.

See *Configuring IAS Security* for information about setting up IAS integration support.

Note

You cannot migrate existing mobile apps that are configured with XSUAA to IAS. IAS is only supported for new mobile apps.

- (Does not appear if IAS is selected) Under [Role Settings](#), view or edit settings. In [Edit Role Settings](#), select [Enable Role Settings](#) to use roles.
Once enabled, you can specify one or more roles, such as Admin or Developer, and click *OK*. If you make any changes, you must restart the application.
- Under [Application Versioning](#), select [Enable Application Versioning](#) to manage versions for this application. Once enabled, you can view and edit versions.


- Click the  icon to add a version.
- Modify the application version settings:

Version Settings


Field	Description
Active	Whether the application version is active and available for use.

Field	Description
Version	Application version, such as 1 . 0, 3 . 3 PL02, and 5 . 2. A version is mandatory.
Check Platforms	Indicates whether the Mobile Services server should check for specific platforms when processing app versions. Set the toggle to <i>Off</i> (default) if the app versions for all platforms should be processed, and set to <i>On</i> if app versions for specific platforms should be processed. Once set to <i>On</i> , select one or more platforms from the list of <i>iOS</i> , <i>Android</i> , and <i>Windows</i> . Note that apps that were introduced before <i>Check Platforms</i> was introduced (Version: 2303) are set to <i>Off</i> .
Description	Application version description, such as BETA, Initial release, and Patch to Version 3.3 - Windows only.
Actions	Actions you can perform, such as edit or delete.


3. Click *OK* to save, or *Cancel*. If you make any changes, you must restart the application.

9. Under *API Key*, if you selected *OAuth*, *SAML*, or *API Key Only* as the Security Configuration, you can view or edit API keys. If you selected *API Key Only*, an API key is created using the current date and time. You must have at least one API key or you receive a warning. Select  to add or delete keys.

Under *Anonymous Access*, view or edit settings. In *Anonymous Access*, select *Allow Anonymous Access* to enable client-side applications to use API keys as credentials.

1. To edit, click the  icon to add an API key.
2. Add one or more valid API keys for the application.

API Key Settings

Field	Description
API Key	When implemented, at least one API key is required.
Creation Date	The server generates the creation date once the API key is created.
Actions	Select  to delete an API key.

3. Click *OK* to save, or *Cancel*. The API keys and their creation dates appear. If you make any changes, you must restart the application.

10. Under *Allowed IP/CIDR*, view the list of allowed Internet Protocol/Classless Inter-Domain Routing (IP/CIDR) addresses, or ranges of addresses. Any request that includes one of these addresses will be performed; requests for addresses that are not in the list will be ignored. You can add, change, and delete entries.

To make entries:

1. Select the edit icon.
 2. In [Edit Allowed IP/CIDR](#), select the plus icon, and then enter an address or range of addresses, for example:
 - 10.0.0.3
 - 10.100.0.1/32
 - 2a0f:ed40::/29
 3. To delete an entry, select [X](#).
 4. To modify an entry, make changes to a current entry.
 5. Select [OK](#) to save changes, and confirm. The change appears in the list.
11. Under [Cross Domain Access](#), view the list of cross-domain access entries. These entries enable application users to access additional security domains based on the values you provide. You can add, change, and delete entries. See also, *Cross-Origin Resource Sharing Requests*.

To make entries:

1. Select the [+](#) icon to add an entry.
2. In [Create Cross Domain Access](#), enter the values, and.

Cross Domain Access Properties

Properties	Descriptions
URI Pattern	A string of characters that matches the destination.
Origin	<p>A comma-delimited list of URIs that can access the resource, for example, <code>http://example.com</code> or <code>http://*.example.com</code>.</p> <p>The default value "*" means any URI can access the resource.</p> <p>An empty value prevents cross-origin resource sharing; only URIs in the same origin can access the resource.</p>
Methods	A comma-delimited list of HTTP methods (such as <code>GET</code> and <code>DELETE</code>) that are allowed when accessing the resource.
Max Age	The number of seconds for which the results of a request can be cached. The default is 3600 seconds (60 minutes).
Allow Credential	Always set to On. The server includes cookies when it submits requests.

Properties	Descriptions
Headers	<p>A comma-delimited list of HTTP request headers that you can specify in requests.</p> <div> <p>Note</p> <p>An empty value means any requested headers are accepted.</p> </div> <p>The following headers are usually used in requesting Mobile Services:</p> <ul style="list-style-type: none"> • <code>accept</code> • <code>authorization</code> • <code>maxdataserviceversion</code> • <code>x-smp-appcid</code> • <code>x-smp-appid</code>
Expose Headers	A comma-delimited list of response headers that browsers can access.

3. Select [Save](#). The new entry appears in the list.

12. If you locked the application while making changes, select [Unlock](#) to release it.

1.6.1.1.4 Defining Push Notifications

Configure push-related settings for the selected application.

The push listener service provided with the server allows back-end systems to send native notifications to devices. Users must bind the push feature to the application from the Mobile Services cockpit.

[Android Push Notifications \[page 343\]](#)

To enable client applications to receive Firebase Cloud Messaging (FCM) notifications, configure Android push notifications.

[Apple Push Notifications \[page 344\]](#)

Configure APNs push notifications for the selected iOS client application or SAP Mobile Cards.

1.6.1.1.4.1 Android Push Notifications

To enable client applications to receive Firebase Cloud Messaging (FCM) notifications, configure Android push notifications.

Prerequisites

You can use the HTTP v1 API, which uses a Firebase Service Account Private Key as the credential; or the legacy deprecated HTTP API with Sender ID and Server Key.

Note

Mobile Services cockpit currently supports both methods.

Google has deprecated the legacy HTTP API that uses Server Key authentication and will discontinue the authentication in June 2024. You must update your configuration to Firebase Service Account Private Key beforehand.

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select [Features](#) [Push Notification](#).
3. Under Android select whether to use [Firebase Service Account Private Key](#) or [Server Key](#) for push credentials.

For [Firebase Service Account Private Key](#):


1. In [Service Account Private Key File](#), browse to select the private key file that you downloaded from the Firebase platform.
2. If the file is valid, the remaining fields are populated automatically using values in the private key file.

Firebase Private Key Properties

Properties	Description
Project ID	The project ID named in the private key file.
Private Key ID	The private key ID named in the private key file.
Client Email	The client email named in the private key file.

For [Server Key \(Deprecated\)](#):

Server Key Credentials



Properties	Description
Access Key	The access key you obtained for your Google API project (https://firebase.google.com/docs/cloud-messaging ).
Sender ID	The project identifier.

4. Click [Save](#).




1.6.1.1.4.2 Apple Push Notifications

Configure APNs push notifications for the selected iOS client application or SAP Mobile Cards.

Context

You can use either certificate or token-based authentication for APNs. The advantage of using token-based authentication is that the credentials do not expire automatically, whereas APNs certificates usually expire after one year. You can manually control the lifecycle of the key used for token-based authentication in developer.apple.com . Obtain the credentials used for token-based authentication from your developer account on developer.apple.com .

Procedure

1. In Mobile Services cockpit, select [Mobile Applications](#).
2. Select an application, then select [Mobile Push Notification](#) under [Assigned Features](#) (or add it first); for SAP Mobile Cards select  [Features](#) .
3. To configure APNs for a development and testing environment, select [Sandbox](#) or [Production](#). See [Apple Developer - Certificates](#)  for certificate types and how to retrieve.
 - Select [Sandbox](#) if your mobile app is signed with an iOS Development Certificate.
 - Select [Production](#) if your mobile app is signed with an iOS Distribution Certificate.
 - Select [None](#) if you don't want to receive APNs push notifications.
4. Choose either [Certificate](#) or [Token-Based](#) as the authentication type.

For [Certificate](#):

1. Select [Browse](#) to navigate to the APNs certificate file, select the file, and click [Open](#).
2. Enter a valid password.
3. (Optional) Identify the topic bundle ID that is specific to the iOS application you're setting up to receive notifications. If not configured, the bundle ID is extracted from the certificate.

For [Token-Based](#):

Token Properties

Property	Description
Topic (Bundle ID)	The bundle ID that is specific to the iOS application you're setting up to receive notifications.
Team ID	The 10-character team ID you use for developing your company's apps. Obtain this value from your developer account.
Key ID	The 10-character string that is provided by Apple when you create a key.
Key (P8)	The authentication token signing key (.p8) file that you've downloaded from Apple. Select Browse to navigate to the APNs key file, select the file, and click Open .

5. Save your changes.

1.6.1.1.5 Uploading Client Resources

Upload client resources (or resource bundles) and manage service keys for the selected application.

Context




Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Mobile Services. The administrator can modify the client resource bundle settings in Mobile Services cockpit. Keep in mind these resource bundle guidelines:

- Supportability – the resource bundle can be of any type (.pdf, .xls, .xml, or any other extension), with no restrictions, except that password protected zip files are not supported.
- Size – the resource bundle is restricted in size. The maximum size is 1GB, because Cloud Foundry saves the bundle content in ObjectStore DB. The maximum total bundle count is 100 for an application. For larger files, work with an application developer for performance issues.
- Default resource bundle – the first resource bundle that you upload is considered to be the default. After that, you can upload additional versions of the bundle, but only one can be the default. You can delete obsolete resource bundle versions.
- URL for the default resource bundle – `https://<mobile services host>/bundles/<ApplicationId>/`
Also supports URL – `https://<mobile services host>/mobileservices/application/{<applicationId>}/bundles/v1/runtime/bundle/application/{<applicationId>}`
- URL to access other resource bundles – `https://<mobile services host>/bundles/<ApplicationId>/<BundleName>:<BundleVersion>`
Also supports URL – `https://<mobile services host>/mobileservices/application/{<applicationId>}/bundles/v1/runtime/bundle/application/{<applicationId>}/bundle/{<bundleName>}/version/{<bundleVersion>}`

Note

To find your `<mobile_services_host>` name, open the application *APIs* tab. The Server URL value is also the `<mobile_services_host>` name.

Procedure

1. In Mobile Services cockpit, select **Mobile Applications** > **Native/MDK**.
2. Select an application, then select **Mobile Client Resources** under **Assigned Features** (or add it first).
3. Select **Configuration** to add client resource.
 - a. Click the **Upload Client Resource** icon  to create a new client resource:
 - **Bundle Name** – provide a name to identify the resource.
 - **Version** – provide a version number.
 - **Upload Client Resource** – click **Browse**, select the file, and confirm.
 - a. To define a client resource bundle as the default, select it, and click **Save**.
 - a. (Optional) Choose  or  as required.
4. Select **Service Keys** to see service keys configured for the application feature. See *Service Keys*.
5. Select **Info** to view feature details, and to download mobile service data in JSON format.

1.6.1.2 Application Development using REST API

To access SAP Mobile Services REST services, develop your HTTP client application to use the REST Services API.

[Authenticating User Requests \[page 347\]](#)

For all user requests that require authentication, send the authentication information to SAP Mobile Services. The credentials, which you provide in the header, depend on the type of security configuration. You can also use OAuth tokens or existing session cookies to authenticate user requests.

[Authenticating Server Application Requests \[page 348\]](#)

Several methods are used to authenticate access requests from customer server applications.

[Create Application Connection \[page 348\]](#)

You must explicitly register an application connection (also called the registration ID) with mobile platform.

[Create Application Connection with Capability Handling \[page 348\]](#)

Starting with version SAP Mobile Services 1.3 of the connection service, clients can manage form factors and capabilities in the application connections.

[Manage Application Settings \[page 349\]](#)

Application settings describe the application connection details such as application ID, security configuration, and customization resource.

[Managing Application Versions Using REST APIs \[page 349\]](#)

You can automate application version management by integrating it into your application build or application artifact management processes: use REST APIs to deploy, promote, delete, and view information for application versions.

[External Access to Application Versions \[page 354\]](#)

Provides access to information (meta-data and binary) about published artifacts stored in the App Lab service. These APIs require authentication and authorization by use of a service-key.

[Native Push Notification for a Back End \[page 354\]](#)

The mobile platform uses the native notification mechanisms provided by individual device platforms such as APNs, GCM, BIS/BES, WNS, and MPNS to send notifications. Back-end systems use the Push REST service to notify the mobile platform about any notification messages it sends to devices.

[Registering Clients for Native Push Notifications \[page 386\]](#)

Enable native push notifications, and register your application to receive push notifications.

[Service Document \[page 391\]](#)

Get the service document for the application connection.

[Logging Out Users \[page 392\]](#)

Terminate an active user session with the user logout service. This is useful for multiple user scenarios.

[Storage Service \[page 393\]](#)

As an application developer, you can use REST services to store device application configuration data in the server.

[Client Resources Service \[page 393\]](#)

As an application developer, you can use REST services to download resources from the server.

[Client Log Upload Service \[page 393\]](#)

As an application developer, you can use REST services to upload client logs to the server.

[Push as API Service \[page 393\]](#)

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

[Client Usage Report and User Feedback Upload Service \[page 394\]](#)

The Client Usage Report and User Feedback Upload service allows an application to upload a client usage report and user feedback for SAP Mobile Services.

[Role Service \[page 394\]](#)

As an application developer, you can use REST services to enable mobile devices to retrieve user name and roles associated with the user according to the System for Cross-domain Identity Management (SCIM) protocol. The role service allows you to get the logical roles for the current user, and you can use it to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

1.6.1.2.1 Authenticating User Requests

For all user requests that require authentication, send the authentication information to SAP Mobile Services. The credentials, which you provide in the header, depend on the type of security configuration. You can also use OAuth tokens or existing session cookies to authenticate user requests.

- Basic authentication
The user name and password should be valid for the specified authentication URL.

- HTTP Header Name: Authorization
- HTTP Header Value: Basic <base64 encoded form of username:password>

1.6.1.2.2 Authenticating Server Application Requests

Several methods are used to authenticate access requests from customer server applications.

1. Access mobile services through the mobile-approuter, which is protected by OAuth security type, customer server application has to use a valid user OAuth token issued by the xsuaa service instance of the mobile approuter application. The token might be received by customer server application from mobile client application.
2. Access mobile services through the mobile-approuter, which is protected by Basic security type, customer server application can use a valid basic authorization header (username and password).
3. Access mobile services directly by the service URL, which is fetched from the service binding of mobile approuter application, service key authentication can be used. Note that currently only the push service and cards service support service key.

1.6.1.2.3 Create Application Connection

You must explicitly register an application connection (also called the registration ID) with mobile platform.

You can specify customized application properties for client requests. Provide the application connection ID, `X-SMP-APPCID`, using an explicit request header or a cookie. If the connection ID is missing, mobile platform generates a universally unique ID (UUID), which is communicated to the device through the response header and cookie `X-SMP-APPCID`.

Related Information

[Create Application Connection \[page 402\]](#)

1.6.1.2.4 Create Application Connection with Capability Handling

Starting with version SAP Mobile Services 1.3 of the connection service, clients can manage form factors and capabilities in the application connections.

During registration, the device sends its form factor (such as smartphone or tablet), and the client can send a certain capability name [such as `purchaseOrder-display`, or a wildcard (*) in case the device has all the capabilities]. When the device user adds or removes a capability, the application connection is updated.

Related Information

[Push API Notification Scenarios \[page 357\]](#)

[Push-to-Capability Scenario \[page 367\]](#)

[Create Application Connection with Capability Handling \[page 403\]](#)

1.6.1.2.5 Manage Application Settings

Application settings describe the application connection details such as application ID, security configuration, and customization resource.

1.6.1.2.6 Managing Application Versions Using REST APIs

You can automate application version management by integrating it into your application build or application artifact management processes: use REST APIs to deploy, promote, delete, and view information for application versions.

Note

You must use a secure port and the HTTPS protocol to make REST API calls. See your client documentation for details about how to submit a request over HTTPS to satisfy the server's security requirements.

[Deploying Hybrid Apps Using the REST API \[page 349\]](#)

Deploy a new or updated hybrid app to SAP Mobile Services using the POST application REST API.

[Promoting Hybrid Apps Using the REST API \[page 351\]](#)

Promote a new hybrid app to make it the current version of the application using the PUT application REST API. Only administrators can run this API; developers cannot.

[Retrieving Hybrid App Details Using the REST API \[page 352\]](#)

Retrieve details about a new or current version of a hybrid app using the GET application REST API.

[Deleting Hybrid Apps Using the REST API \[page 352\]](#)

Delete a hybrid app using the DELETE application REST API. Developers can delete only the applications they created.

1.6.1.2.6.1 Deploying Hybrid Apps Using the REST API

Deploy a new or updated hybrid app to SAP Mobile Services using the POST application REST API.

Note

You cannot deploy a hybrid app for a specific platform: everything in the Kapsel application file is deployed. Once the application is deployed, you can promote or delete hybrid apps for specific platforms as needed.

To attach a Kapsel application file as a parameter in a REST client, use the `curl` command line tool.

After an application is deployed, it is considered to be a new version. You can activate it by promoting it to the current version, which allows users to download patches and upgrade the application on their devices.

Syntax

Send a POST request to the following URI:

```
https://<mobile_services_host>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>
```

Note

To authenticate, specify the user name and password in each request.

Returns

A response with information about the new and current version of the application. For example:

```
{ "newVersion":  
  { "requiredKapselVersion": "1.5",  
    "developmentVersion": "1.2.5",  
    "description": "An update for the sample app.",  
    "revision": -1 },  
  "currentVersion":  
    { "requiredKapselVersion": "1.5",  
      "developmentVersion": "1.2.4",  
      "description": "A sample app.",  
      "revision": 2 }  
}
```

If successful, a 200 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-  
server.pem> --X POST -i https://service-cockpit-  
web-development.cfapps.sap.hana.ondemand.com/cockpit/v1/org/mobiledev_mobile-ops-  
subscriber/space/development-qa/app/MyTestAppId/service/app-update/Admin/kapsel/  
jaxrs/KapselApp/MyTestAppId
```

1.6.1.2.6.2 Promoting Hybrid Apps Using the REST API

Promote a new hybrid app to make it the current version of the application using the PUT application REST API. Only administrators can run this API; developers cannot.

Promote a hybrid app for a specific platform or for all platforms.

Note

To authenticate, specify the user name and password in each request.

Syntax

To promote a hybrid app for all platforms, send a PUT request to the following URI:

```
https://<host>:<admin_port>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>
```

To promote a hybrid app for a specific platform, send a PUT request to the following URI:

```
https://<host>:<admin_port>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>/<action>
```

Where **<action>** is one of:

- stage
- unstage
- promotePending
- promoteStage

After the application is promoted, users can upgrade the application on their devices.

Returns

If successful, a 200 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-  
server.pem> --X PUT -i https://service-cockpit-  
web-development.cfapps.sap.hana.ondemand.com/cockpit/v1/org/mobiledev_mobile-ops-
```

```
subscriber/space/development-qa/app/MyTestAppId/service/app-update/Admin/kapsel/jaxrs/KapselApp/MyTestAppId/promotePending
```

1.6.1.2.6.3 Retrieving Hybrid App Details Using the REST API

Retrieve details about a new or current version of a hybrid app using the GET application REST API.

Syntax

Send a GET request to the following URI:

```
https://<mobile_services_host>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>
```

Note

To authenticate, specify the user name and password in each request.

Returns

If successful, a 200 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-server.pem> --X GET -i https://service-cockpit-web-development.cfapps.sap.hana.ondemand.com/cockpit/v1/org/mobiledev_mobile-ops-subscriber/space/development-qa/app/MyTestAppId/service/app-update/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```

1.6.1.2.6.4 Deleting Hybrid Apps Using the REST API

Delete a hybrid app using the DELETE application REST API. Developers can delete only the applications they created.

Delete a hybrid app from a specific platform or from all platforms.

Note

To authenticate, specify the user name and password in each request.

Syntax

To delete a hybrid app from all platforms, send a DELETE request to the following URL:

```
https://<mobile services host>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>
```

To delete a hybrid app from a specific platform, send a DELETE request to the following URL:

```
https://<mobile services host>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>/<PLATFORM>
```

To delete pending revisions they created, developers can:

1. Add the X-HTTP-METHOD=DELETE parameter to the request header.
2. Format the request body as this JSON string:

```
[{"platform": "android", "revisions": [1, 2, 3]},  
{"platform": "ios", "revisions": [4, 5, 16]}]  
[{"platform": "android", "revisions": [2]}, {"platform": "ios"}]  
[{"platform": "android"}]
```

3. Send a POST request to the following URL:

```
https://<mobile services host>/cockpit/v1/org/<ORG_NAME>/space/<SPACE_NAME>/app/  
<APP_ID>/service/app-update/Admin/kapsel/jaxrs/KapselApp/<APP_ID>/deletePendings
```

Returns

If successful, a 200 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-  
server.pem> --X DELETE -i https://service-cockpit-  
web-development.cfapps.sap.hana.ondemand.com/cockpit/v1/org/mobiledev_mobile-ops-  
subscriber/space/development-qa/app/MyTestAppId/service/app-update/Admin/kapsel/  
jaxrs/KapselApp/MyTestAppId
```

1.6.1.2.7 External Access to Application Versions

Provides access to information (meta-data and binary) about published artifacts stored in the App Lab service. These APIs require authentication and authorization by use of a service-key.

1.6.1.2.8 Native Push Notification for a Back End

The mobile platform uses the native notification mechanisms provided by individual device platforms such as APNs, GCM, BIS/BES, WNS, and MPNS to send notifications. Back-end systems use the Push REST service to notify the mobile platform about any notification messages it sends to devices.

Request

URL: `https://<mobile services host>/restnotification/<registration ID>`

HTTP Method: `POST`

Request Parameters

Parameter	Type	Description
restnotification	Mandatory	Received from the proxy push endpoint.
registration ID	Mandatory	Is sent to the device when a user registers and connects to the application from the device.
gcmNotification	Device specific	Used for communication between SAP Mobile Services and the GCM server, but provides notification elements used to send GCM push requests to SAP Mobile Services.
wnsNotification	Device specific	Used for communication between SAP Mobile Services and the WNS server, but provides notification elements used to send WNS push requests to SAP Mobile Services.

You can also send notification data using URL arguments.

[Notification Data Sent Using Push API \[page 355\]](#)

A push message appears as a notification on a device, informing the user of an action he or she must take. In order to send push notifications to an application, you must have Notification User privilege assigned to your user ID.

[Push Notification JSON Payload Handling \[page 369\]](#)

SAP Mobile Services provide push notification JSON payload handling for APNs, GCM, and WNS. When enabled, the mobile platform maps JSON payload values to custom platform values.

[Notification Data Sent Through HTTP Headers \[page 381\]](#)

Notification data can be sent by the back end as generic HTTP headers or as device platform-specific HTTP headers.

[SAP Gateway Notification Support \[page 383\]](#)

There are no specific handling requirements for sending notifications on the SAP Gateway side. Mobile Services sends notifications using gateway-specific headers.

[Notification Sent in URL Format \[page 384\]](#)

Notification data can also be sent by using the REST client, using URL arguments as part of the mobile platform push endpoint, or as the delivery address URL.

Related Information

[Set Up Customer Accounts \[page 20\]](#)

[Notification Sent in URL Format \[page 384\]](#)

[Notification Data Sent Using Push API \[page 355\]](#)

1.6.1.2.8.1 Notification Data Sent Using Push API

A push message appears as a notification on a device, informing the user of an action he or she must take. In order to send push notifications to an application, you must have Notification User privilege assigned to your user ID.

Configuring Authentication Provider Settings in Mobile Services Cockpit (SAP BTP)

By default, SAP BTP uses SAP Cloud ID service to authenticate users against SAP user accounts. Assign the `Notification User` role to the SAP user ID to be able to send the push notification to the device.

1. Open the SAP BTP Cockpit
2. Select [Services](#).
3. Under Mobile Services, select [Roles](#).
4. Select the [Notification User](#) role and assign user ID to the role.

Note

In SAP Mobile Services, an unauthenticated user is referred to as a public user rather than as a `nosec_identity` user as in the on-premise version of the server.

Testing Notification Service

You can use any REST tool, such as “Advanced Rest Client” or “Postman”, available from the Google Chrome Web store for testing.

The `restnotification` API sends native push notifications to the applications. This RESTful service provides more flexibility for sending push messages than existing interfaces that are based on HTTP headers or URL parameters. Earlier push interfaces required that you send messages to a registration ID. The `restnotification` interface also sends the message to a specific user or to all users of a specific application.

The `restnotification` API sends messages to multiple recipients. The messages are queued in the server and sent out asynchronously.

Request

URL: `https://<mobile_services_host>/restnotification/<resource>`

HTTP Method: *GET*

[Push API Notification Scenarios \[page 357\]](#)

Send push notifications to devices that are registered to an application.

[Push-to-Capability Scenario \[page 367\]](#)

The push-to capability scenario is a push notification variation. This scenario enables you to push notifications to applications that have certain capabilities rather than to individual applications.

Related Information

[Native Push Notification for a Back End \[page 354\]](#)

1.6.1.2.8.1.1 Push API Notification Scenarios

Send push notifications to devices that are registered to an application.

Request

URL: `https://<mobile services host>/restnotification/application/<applicationId>`

Request Parameters

Parameter	Type	Description
applicationId	Mandatory	ID that uniquely identifies an application.

Request Body Example

```
> POST /restnotification/application/123456789 HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json; charset=utf-8
> Content-Length: 127
> {
  "alert": "alertval",
  "badge": 1,
  "data": "testData",
  "sound": "soundval"
}
< HTTP/1.1 201 Created
< Content-Length: 0
< Date: Mon, 05 May 2014 00:29:38 GMT
< Server: SAP
```

In this scenario, a status code 201 indicates that the server accepts the push notification request. The server forwards these requests to the external push service such as GCM, BES, BIS, APNs, WNS and so on. The status code does not indicate that the server has successfully delivered the notification to the devices.

Response

Other possible HTTP status codes, you may encounter:

Code	Description
400 Bad Request	The request is invalid. Verify the request body.
401 Unauthorized	Authentication is required. No or incorrect credentials were provided. Enter the correct credentials.

Code	Description
403 Forbidden	The user who issued the request does not have the required privileges. Ensure that the user is assigned to the Notification User role.

Users and Devices

To send push notification to all the devices registered to a particular user, use:

URL: `https://<mobile services host>/restnotification/application/<applicationId>/user/<userID>`

Request Body Example

```
> POST /restnotification/application/123456789/user/timmitester HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {
  "alert": "alertval",
  "badge": 1,
  "data": "testData",
  "sound": "soundval"
}
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIDSSO=C05E58BE3CFC685ABB945D53C2AF14FD; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSID=4CC5BC2943E5D3A9B5D924888FC28CB060034F0092911A66B9F079047077798C; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:32:35 GMT
< Server: SAP
```

Registration ID

To send push notification to a device by using an application registration ID, use:

URL: `https://<mobile services host>/restnotification/registration/<applicationRegistrationId>`

Request Body Example

```
> POST /restnotification/registration/9f847e51-3242-4899-9193-39b6e840d657 HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {
```

```

    "alert": "alertval",
    "badge": 1,
    "data": "testData",
    "sound": "soundval"
  }
}
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIONIDSSO=D541E4898186AB304F506D13C0C3F1D0; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSIONID=FDB39F9BAE8A6E1AD4373765A58E094A14B8FDFB8289CC70E51B77A284C50736; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:36:15 GMT
< Server: SAP

```

Users per Application

To send push notification to all the users of an application, use:

URL: `https://<mobile services host>/restnotification/application/<applicationId>/user`

Request Body Example

```

POST /restnotification/application/123456789/user HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json; charset=utf-8
> Content-Length: 277
> {
  "notification": {
    "alert": "alertval",
    "badge": 1,
    "data": "testData",
    "sound": "soundval"
  },
  "users": [
    "timmitester",
    "user1",
    "user2"
  ]
}
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIONIDSSO=9AD06173C8AB9FC05FD6AA8DC55BB9AE; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSIONID=DFB2D2AC4EBAA4903553EB7C5A0C90870BD4B8F3A3DC19A5FD984673EB1BD646; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:38:32 GMT
< Server: SAP

```

Registration List

To send push notifications to a list of registrations, use:

URL: `https://<mobile services host>/restnotification/registration/`

Request Body Example

```
POST /restnotification/registration HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 466
> {
  "notification": {
    "alert": "alertval",
    "badge": 1,
    "data": "testData",
    "sound": "soundval"
  },
  "registrations": [
    "3078e166-f144-4288-9dbc-1d192afe18d8",
    "9f847e51-3242-4899-9193-39b6e840d657",
    "4d1ccdf9-058a-42cf-a625-c4ed48944729",
    "f05dc905-b859-45fa-afdc-da3b630d2b48",
    "282be579-783e-40fb-b376-25bed5e13606"
  ]
}
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SSIDSSO=BCA5FCB41DD7F451410E3E8BB59E8F7A; Path=/; HttpOnly
< Set-Cookie: X-SMP-SSID=2AC74022B258178ED3A88E4B2FA10AB41093F53C3D0A77976FE6FE076F1E3CC2; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:41:52 GMT
< Server: SAP
```

Capability

Use `capability` to identify device capabilities. This enables you to push notifications to applications with specific capabilities rather than to individual applications. To send push notifications to applications that support specific capabilities use:

URL: `https://<mobile services host>/restnotification/capability/<capabilityName>/`

Capability supports two modes:

- **Wildcard (*):** the device has all capabilities. When a push notification is sent, the device form factor must match.
For example, Jean registers a device with a wildcard capability `capabilityName: *` and `form factor: tablet`, and Jake registers with `capabilityName: *` and `form factor: phone`. When the notification `capability: 'purchaseOrder-display'` and `form factor: phone` is pushed to both users, only Jake gets the notification. Jean does not get the notification, because the form factor does not match.
- **Match capability name only:** the device has a certain capability name. When a push notification is sent, the notification must match the capability, and the form factor is ignored.
For example, Yijie registers a device with a specific capability name `capability: 'purchaseOrder-display'` and `form factor: phone`. When a notification is pushed to `capability: 'purchaseOrder-display'` and `formFactor: tablet`, Yijie receives the notification because the capability matches. The form factor `formFactor: tablet` is ignored.

Note

You can use `CapabilityName` either as a wildcard (*), or as specific strings, but not as a string + wildcard (*), such as `purchase*`. For example, if you set `CapabilityName=purchase*` using the REST client, and then send a notification to `purchaseOrder-display`, the device does not get the notification.

Request Body Example

```
POST http://localhost:8082/restnotification/capability/display HTTP/1.1
> Accept: application/json
> Authorization: Basic cHVsaDpzZWNyZXQ=
>
{
  "notification": {
    "data": "{\n\"NotificationId\": \"005056AB5B8D1ED4B99CC017A78D2429\", \"Text\": \"You have a new purchase order for approval\", \"NavigationTargetObject\": \"purchaseOrder\", \"NavigationTargetAction\": \"display\", \"NavigationTargetParam\": [{\n\"Key\": \"ID\", \"Value\": \"4711\"}], \"Actions\": [{\n\"ActionId\": \"approve\", \"ActionText\": \"Approve\", \"BulkActionText\": \"Approve all\", \"Nature\": \"POSITIVE\"}, {\n\"ActionId\": \"reject\", \"ActionText\": \"Reject\", \"BulkActionText\": \"Reject all\", \"Nature\": \"NEGATIVE\"}], \"NotificationTypeId\": \"purchaseOrder\"}]",
    "alert": "You have a new purchase order for approval",
    "sound": "beep",
    "customParameters": {
      "apns.category": "INVITE_CATEGORY"
    }
  },
  "users": [
    {
      "badge": 3,
      "formFactor": [
        "tablet",
        "smartphone"
      ],
      "user": "john"
    },
    {
      "badge": 2,
      "formFactor": [
        "smartphone"
      ],
      "user": "jane"
    }
  ]
}
< HTTP/1.1 201 Created
< Content-Type: application/json
```

Response Body

```
> {
  "status": {
    "value": "OK",
    "code": 0
  },
  "results": [
    {
      "status": {
        "value": "OK",
        "code": 0
      },
      "registrationId": "00783d2a-7031-49d0-bc3b-1aae90772955"
    },
    {
      "status": {

```

```

        "value": "OK",
        "code": 0
    },
    "registrationId": "3403fdc4-9ecb-48e5-8e11-b2ac99ab0e90"
}
]
}

```

Capability (Enhanced Badge Handling)

There is not always a one-to-one relationship between the capability that is pushed to, and the badge number for the relevant mobile app. A single mobile app could be capable of handling multiple different capabilities. However, this information is only known by SAP Mobile Services, not by the back end. Consequently, the back end cannot deliver the accurate badge number for a random mobile application, it can only deliver badge numbers per capability. Typically the API only sends a single badge number, which is too restrictive for this scenario, and may result in inaccurate badge numbers showing up on a user's device.

A single push would still be targeted towards a particular capability, but the capability-based push API has been enhanced so that the back end can send the badge number per capability. SAP Mobile Services calculates the actual badge number for an application based on its registered capabilities, and sends the sum of all individual badge values to the device.

```

POST URL:
https://<mobile services host>/restnotification/capability/capability2
Payload:
{
  "notification": {
    "data": "...",
    "alert": "You have a new purchase order for approval",
    "customParameters": {
      "apns.category": "action"
    }
  },
  "users": [
    {
      "badge": 3,
      "formFactor": [
        "tablet"
      ],
      "user": "jean"
    },
    {
      "badges": {
        "capability1": 2,
        "capability2": 4,
        "capability3": 8
      },
      "formFactor": [
        "phone"
      ],
      "user": "jake"
    }
  ]
}

```

The badge property can either be a plain number, which is sent as is to the native push service (the default implementation); or alternatively the badges property can contain a map of capability/badge pairs (the enhanced implementation). The actual badge value is then determined based on the target application's registered capabilities.

In the example above, the push is delivered to the mobile apps that "jake" has installed, which have registered for "capability2". Assuming "jake" has one app with the wildcard capability, and one app with capabilities "capability2" and "capability3"; the badge count for the app with wildcard capability would be 14, the badge count for the other app would be 12.

The API enables a badge to be reset without sending an actual notification, and allows the notification data and alert to be empty. In cases where capability context is not provided, the API supports the wildcard "*" as a capability in the URL. In other words, a push to `/restnotification/capability/*` must be delivered to all apps for a user that has registered with any capability (or the wildcard capability), and all other matching rules must be obeyed, for example, the form factor must still match.

Customize Push Notification Types

Use `customParameters` to override the value for a particular notification type. Customize push notification types-alert, badge, data, and sound in the payload by prefixing with:

- apns
- bbbis
- bbbes
- gcm
- mpns
- wns

HTTP Method: *POST*

Example: Push to all users to the application with application ID "XYZ". Issue a POST method on:

```
> POST /restnotification/application/XYZ HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> {
  "alert": "alertval",
  "badge": 1,
  "data": "testData",
  "sound": "soundval"
}
```

Example: To reset or override the value of the notification type parameter - `sound` in your Android device, you can use the `customParameters` to override the value of the sound parameter:

```
> {
  "alert": "alertval",
  "badge": 1,
  "customParameters": {
    "gcm.sound": "soundforgcm"
  },
  "data": "testData",
  "sound": "soundval"
}
```

Category (APNs)

Use `category` for "actionable" APNs push notifications. These notifications can be sent through SAP Mobile Platform and SAP Mobile Services directly, or through Push Hub.

Example: REST(ful) push request containing a JSON payload. The category is a sub-element of the `customParameters` element called "apns.category". Issue a POST method on:

```
> POST /restnotification/registration/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/json
> Authorization: Basic cHVzaDpzZWNyZXQ=
> Content-Length: 117
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
> {
  "alert": "PushAlert",
  "data": "pushTest",
  "customParameters":
    { "apns.category": "SoapUICategory" }
}
```

Example: non-SAP Gateway notification. Include the header "X-SMP-APNS-CATEGORY". Issue a POST method on:

```
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> X-SMP-APNS-CATEGORY: SoapUICategory
> X-SMP-APNS-DATA: pushTest
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: SAP Gateway notification Include the header "X-SAP-POKE-CATEGORY". Issue a POST method on:

```
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> X-SAP-POKE-DATA: pushTest
> X-SAP-POKE-CATEGORY: SoapUICategory
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: URL parameter encoded. Append the parameter "category=" to the request URL. Issue a POST method on:

```
> POST /Notification/<<applicationId>>?
  alert=PushAlert&data=pushTest&category=SoapUICategory HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Content Available (APNs)

Use `contentAvailable` for "actionable" APNs push notifications, which enable users to take action without changing focus. These notifications can be sent through SAP Mobile Platform and SAP Mobile Services directly, or through Push Hub.

Example: REST(ful) pushPush request containing a JSON payload. The `content-available` field is a sub-element of the `customParameters` element called "apns.contentAvailable", and is of type "boolean". Issue a POST method on:

```
> POST /restnotification/registration/<<applicationId>> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/json
> Authorization: Basic cHVzaDpzZWNYZXQ=
> Content-Length: 146
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
> {
  "data": "pushTest",
  "customParameters":
    { "apns.contentAvailable": "true" }
}
```

Example: non-SAP Gateway notification. Include the header "X-SMP-APNS-CONTENT-AVAILABLE". Issue a POST method on:

```
> POST /Notification/<<applicationId>> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNYZXQ=
> X-SMP-APNS-CONTENT-AVAILABLE: true
> X-SMP-APNS-DATA: pushTest
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: SAP Gateway notification Include the header "X-SAP-POKE-CONTENT-AVAILABLE". Issue a POST method on:

```
> POST /Notification/<<applicationId>> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNYZXQ=
> X-SAP-POKE-DATA: pushTest
> X-SAP-POKE-CONTENT-AVAILABLE: true
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: URL parameter encoded. Append the parameter "Content_Available=" to the request URL. Issue a POST method on:

```
> POST /Notification/<<applicationId>>?
  alert=PushAlert&data=pushTest&Content_Available=true HTTP/1.1
```

```
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNYZXQ=
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Sound (APNs)

Use sound to submit the name of the sound file to be played along with a push message when using the gateway header. This header can hold an arbitrary string representing the name of the sound file, which is appended to the APNs message's sound parameter during the process of building the APNs payload. To play the default sound the user has configured for notifications, set the parameter to the value "default": "sound" : "default".

Example: the RESTful SAP Gateway notification includes the "X-SAP-POKE-SOUND" header in the request.

```
POST http://localhost:8080/Notification/fa8ed84c-9033-45c0-a0a0-971a2d217367
HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/xml
Authorization: Basic ****
X-SAP-POKE-SOUND: SomeSoundFile
X-SAP-POKE-TITLE: Alert from SMP3
Content-Length: 0
Host: localhost:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Request Body Example

```
> POST /restnotification/application/123456789 HTTP/1.1
> Authorization: Basic cHVzaDpzZWNYZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {
  "alert": "alertval",
  "badge": 1,
  "data": "testData",
  "sound": "soundval"
}
< HTTP/1.1 201 Created
< Content-Length: 0
< Date: Mon, 05 May 2014 00:29:38 GMT
< Server: SAP
```

In this scenario, the "soundval" file is associated with the request.

Example outgoing payload to APNs:

```
{"aps":{"alert":"Alert from SMP3","sound":"SomeSoundFile"}}
```

Related Information

[Push-to-Capability Scenario \[page 367\]](#)

[Create Application Connection with Capability Handling \[page 403\]](#)

[Create Application Connection with Capability Handling \[page 348\]](#)

1.6.1.2.8.1.2 Push-to-Capability Scenario

The push-to capability scenario is a push notification variation. This scenario enables you to push notifications to applications that have certain capabilities rather than to individual applications.

Applications provide capability information, such as 'purchaseOrder-display', when they register an application connection. The platform then uses the capability information to push notifications to the device.

Usage

Two supported modes:

- Wildcard (*): the device has all capabilities; notifications must match the device's form factor.
- Match capability name only: the device has a certain capability name; notifications must match the capability, and the form factor is ignored.
Enhanced badge handling is supported as part of capabilities. The mobile app handles multiple different capabilities, and the sum of all individual badge values is sent to the device.

Wildcard (*)

A wildcard indicates that the device has all capabilities. When someone sends a notification to a certain capability name, then the device form factor must match.

For example, Jean registers a device with a wildcard capability `capabilityName: *` and `form factor: tablet`, and Jake registers with `capabilityName: *` and `form factor: phone`. When the notification `capability: 'purchaseOrder-display'` and `form factor: phone` is pushed to both users, only Jake gets the notification. Jean does not get the notification, because the form factor does not match.

```
POST URL:
https://<mobile services host>/restnotification/capability/purchaseOrder-display
Payload:
{
  "notification": {
    "data": "{ \"NotificationId\": \"005056AB5B8D1ED4B99CC017A78D2429\", \"Text\": \"You have a new purchase order for approval\", \"NavigationTargetObject\": \"purchaseOrder\", \"NavigationTargetAction\": \"display\", \"NavigationTargetParam\": [{ \"Key\": \"ID\", \"Value\": \"4711\" }], \"Actions\": [{ \"ActionId\": \"approve\", \"ActionText\": \"Approve\", \"BulkActionText\": \"Approve all\", \"Nature\": \"POSITIVE\" }, { \"ActionId\": \"reject\", \"ActionText\": \"Reject\", \"BulkActionText\": \"Reject all\", \"Nature\": \"NEGATIVE\" } ], \"NotificationTypeId\": \"purchaseOrder\" }",
    "alert": "You have a new purchase order for approval",
    "customParameters": {
      "apns.category": "action"
    }
  }
}
```

```

    },
    "users": [{
      "badge": 3,
      "formFactor": ["tablet"],
      "user": "jean"
    }, {
      "badge": 3,
      "formFactor": ["phone"],
      "user": "jake"
    }
  ]
}

```

Match Capability Name Only

A specific capability name indicates that the notification must match the capability, and that the form factor is ignored.

For example, Yijie registers a device with a specific capability name `capability: 'purchaseOrder-display'` and form factor: `phone`. When a notification is pushed to `capability: 'purchaseOrder-display'` and `formFactor: tablet`, Yijie receives the notification because the capability matches. The form factor `formFactor: tablet` is ignored.

Note

You can use `CapabilityName` either as a wildcard (*), or as specific strings, but not as a string + wildcard (*), such as `purchase*`. For example, if you set `CapabilityName=purchase*` using the REST client, and then send a notification to `purchaseOrder-display`, the device does not get the notification.

Enhanced Badge Handling

There is not always a one-to-one relationship between the capability that is pushed to, and the badge number for, the relevant mobile app. A single mobile app could handle multiple different capabilities. However, this information is only known by SAP Mobile Services, and not by the back end. Consequently, the back end cannot deliver the accurate badge number for a random mobile application; it can only deliver badge numbers per capability. Typically, the API sends only a single badge number, which is too restrictive for this scenario, and may result in inaccurate badge numbers showing up on a user's device.

A single push is still targeted toward a particular capability, but the capability-based push API has been enhanced so that the back end can send the badge number per capability. SAP Mobile Services calculates the actual badge number for an application based on its registered capabilities, and sends the sum of all individual badge values to the device.

```

POST URL:
https://<mobile services host>/restnotification/capability/capability2
Payload:
{
  "notification": {
    "data": "...",
    "alert": "You have a new purchase order for approval",
    "customParameters": {
      "apns.category": "action"
    }
  },
  "users": [
    {
      "badge": 3,
      "formFactor": [
        "tablet"
      ]
    }
  ]
}

```



```

    "user": "jean"
  },
  {
    "badges": {
      "capability1": 2,
      "capability2": 4,
      "capability3": 8
    },
    "formFactor": [
      "phone"
    ],
    "user": "jake"
  }
]
}

```

The badge property can either be a plain number, which is sent as-is to the native push service (the default implementation); or alternatively, the badges property can contain a map of capability/badge pairs (the enhanced implementation). The actual badge value is then determined based on the target application's registered capabilities.

In the example above, the push is delivered to the mobile apps that "jake" has installed, which have registered for "capability2". Assuming "jake" has one app with the wildcard capability, and one app with "capability2" and "capability3"; the badge count for the app with wildcard capability is 14, the badge count for the other app is 12.

The API enables a badge to be reset without sending an actual notification, and allows the notification data and alert to be empty. If capability context is not provided, the API supports the wildcard "*" as a capability in the URL. In other words, a push to `/restnotification/capability/*` must be delivered to all apps for a user that has registered with any capability (or the wildcard capability), and all other matching rules must be obeyed, for example, the form factor must still match.

Related Information

[Push API Notification Scenarios \[page 357\]](#)

[Create Application Connection with Capability Handling \[page 403\]](#)

[Create Application Connection with Capability Handling \[page 348\]](#)

1.6.1.2.8.2 Push Notification JSON Payload Handling

SAP Mobile Services provide push notification JSON payload handling for APNs, GCM, and WNS. When enabled, the mobile platform maps JSON payload values to custom platform values.

[APNs Push Notification Payload Handling \[page 370\]](#)

APNs provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices. The mobile platform maps the incoming JSON payload to APNs custom values.

[FCM Push Notification Payload Handling \[page 372\]](#)

FCM provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

[WNS Push Notification Payload Handling \[page 377\]](#)

WNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

1.6.1.2.8.2.1 APNs Push Notification Payload Handling

APNs provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices. The mobile platform maps the incoming JSON payload to APNs custom values.

Payload Handling for APNs

This section provides examples of JSON payload handling for APNs custom values.

data

The mobile platform maps `data` to an APNs custom value using `key=data`. The value must be a String. For example, the client sends:

```
{ "data" : "{ \"acme1\": \"bar\", \"acme2\" : [ \"bang\", \"whiz\" ] }" }
```

The mobile platform forwards this JSON to APNs:

```
{ "data": "{ \"acme1\": \"bar\", \"acme2\" : [ \"bang\", \"whiz\" ] }" }
```

alert

The mobile platform maps `alert` to NS custom value `aps.alert`. The value of `alert` can be either a String, or a JSON Object converted to a String.

alert Value as a String

The mobile platform sets `alert` as value of `aps.alert`.

alert Value as a JSON Object Converted to a String

The mobile platform parses the String and converts it to a JSON Object. It supports all keys currently supported, such as `body`, `title-loc-key`, `title-loc-args`, `action-loc-key`, `loc-args`, `launch-image`.

Note

The key `body` has a higher priority than `alter_body`. If `body` is included, then `alter_body` is ignored. If only `alter_body` is included, it is used.

For example:

```
{
  "alert": "{ \"alert_body\": \"ignoreThis\", \"body\": \"Your vacation request
has been approved.\", \"action-loc-key\": \"VIEW\", \"loc-key\" :
\"GAME_PLAY_REQUEST_FORMAT\", \"loc-args\" : [ \"Jenna\", \"Frank\" ],
```

```
\\"launch-image\\": \\"push_icon.png\\",\\"title\\" : \\"GameRequest\\",\\"title-loc-key\\" :\\"titleLocKeyVal\\", \\"title-loc-args\\": [\\"foo\\",\\"bar\\"]}"
}
```

is forwarded as:

```
{
  "aps": {
    "alert": {
      "body": "Your vacation request has been approved.",
      "title": "Game Request",
      "launch-image": "push_icon.png",
      "loc-args": [
        "Jenna",
        "Frank"
      ],
      "title-loc-args": [
        "foo",
        "bar"
      ],
      "action-loc-key": "VIEW",
      "loc-key": "GAME_PLAY_REQUEST_FORMAT",
      "title-loc-key": "titleLocKeyVal"
    }
  }
}
```

sound

The mobile platform maps `sound` to the APNs custom value `aps.sound`. The value must be a String. For example:

```
{ "sound": "chime.aiff" }
```

is forwarded as:

```
{ "aps": { "sound": "chime.aiff" } }
```

badge

The mobile platform maps `badge` to the APNs custom value `aps.badge`. The value must be a non-negative Number. For example:

```
{ "badge": 1 }
```

is forwarded as:

```
{ "aps": { "badge": 1 } }
```

apns.category (as a child of "customParameters")

The mobile platform maps `apns.category` to the APNs custom value `aps.category`. The value must be String. When the client sends the following payload:

```
{ "alert": "New vacation request", "customParameters": { "apns.category": "NEW_MESSAGE_CATEGORY" } }
```

it is forwarded as:

```
{ "aps": { "category": "NEW_MESSAGE_CATEGORY", "alert": "New vacation request" } }
```

apns.contentAvailable (as a child of "customParameters")

The mobile platform maps `apns.contentAvailable` to the APNs custom value `aps.contentAvailable`. The value must be either 0 or 1. When the client sends the following payload:

```
{ "customParameters": { "apns.contentAvailable": true } }
```

it is forwarded as:

```
{ "aps": { "content-available": 1 } } - "apns.customValues" as child from "customParameters"
```

The mobile platform maps it to APNs custom values. The value must be a JSON Object converted to String. For example, when the client sends the following payload:

```
{ "alert": "New vacation request", "customParameters": { "apns.customValues": "{ \"acme1\": \"bar\", \"acme2\" : [ \"bang\", \"whiz\" ] }" } }
```

it is forwarded as:

```
{ "aps": { "alert": "New vacation request", "acme2": [ "bang", "whiz" ], "acme1": "bar" } }
```

1.6.1.2.8.2.2 FCM Push Notification Payload Handling

FCM provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

Payload Handling for FCM

This provides information and examples of JSON payload handling for FCM custom values. Notification elements send FCM push requests to the mobile platform.

Note that two properties are available, depending on the push API used:

- `gcmNotification` – used with the older REST notification API of the form: `https://<mobile services host>/restnotification/`. Examples are provided in following sections.
- `gcm` – used with the newer approach used in Cloud Foundry, which implements the back-end push API of the form `https://{servicepath}/mobileservices/push/v1/backend/`. See [Sending Push Notifications via REST API](#) for information. Note that the `gcmNotification` property is ignored by the back-end push API.

JSON Payload Support for FCM Push Notifications

When sending push notifications to FCM, JSON payload handling is transparent to the client since it affects only the communication between the mobile platform and the FCM server. Use these notification elements to send FCM push requests to the mobile platform:

- **priority** – allows the definition of a priority that is forwarded to the respective notification type handler. In case of FCM, the values "normal", which is the default if the field is left blank, and "high" are valid values.
- **gcmNotification** – holds FCM-specific notification elements. Unless specified, the data type for each element is "string". These are optional.
 - collapseKey
 - delayWhileIdle – holds a Boolean value
 - timeToLive – holds a long value (TTL in milliseconds)
 - restrictedPackageName
 - title – if set, this field takes priority over the "alert" field of the notification element
 - body
 - icon
 - sound – if set, this field takes priority over the "sound" field of the notification element
 - tag
 - color
 - clickAction
 - bodyLockKey
 - bodyLocArgs – must hold a serialized JSON list of values
 - titleLockKey
 - titleLocArgs – must hold a serialized JSON list of values

If the JSON payload handling for GCM custom values feature is inactive, the old GCM notification handling behavior is applied, meaning that the above-mentioned notification elements are not part of the push message sent to GCM, and the message sent to GCM remains in plain-text format, instead of JSON format. Through the feature flag, the mobile platform GCM notification handler stays backward compatible, offering the possibility of including the (old) notification payload via headers and URL parameters.

The following example shows a complete push request to the mobile platform, using every element available:

```
{
  "alert": "Updates Available",
  "badge": 1,
  "data": "{\"version\":\"1.13\",\"size\":\"14MB\"}",
  "priority": "high",
  "sound": "DefaultNotificationSound",
  "gcmNotification": {
    "title": "The Title For The App",
    "icon": "TheIcon",
    "body": "The Notification Body",
    "sound": "OverrideSound",
    "color": "Blue",
    "tag": "TheTag",
    "collapseKey": "TheCollapseKey",
    "delayWhileIdle": true,
    "timeToLive": 10,
    "restrictedPackageName": "com.sap.test",
    "clickAction": "TheClickAction",
    "bodyLockKey": "message",
    "bodyLocArgs": "[\"msg1\", \"msg2\"]",

```

```

    "titleLocKey": "titleMessage",
    "titleLocArgs": "[\"tmsg1\", \"tmsg2\"]"
  },
  "customParameters": {
    "gcm.badge": 2
  }
}

```

Notification Element Validation

In earlier versions, only very basic notification element validation was applied, verifying only that at least one of the following elements "alert", "badge" or "data" contained information.

This behavior has been changed to accommodate the "send-to-sync" functionality of GCM, and to provide more granular methods of validating notification payload that is tailored to a specific notification type. See *Send-to-Sync*, below.

The input validation is still performed synchronously, meaning a notification with invalid values is not accepted, and results in an immediate error response being returned. Summary of the latest changes to the mobile platform GCM notification handler:

- **timeToLive** – validation ensures that the value of the field is in the range of 0 to 2419200.
- **priority** – validation ensures that the field contains either the value "normal" or "high", or no value at all, for which GCM assumes the default value of "normal".
- **bodyLocArgs** – validation ensures that the value of this field is a proper serialized JSON list containing only string values by trying to deserialize the JSON string into a Java list of string objects.
- **titleLocArgs** – validation ensures that the value of this field is a proper serialized JSON list containing only string values by trying to deserialize the JSON string into a Java list of string objects.
- **maximum payload size** – according to GCM documentation, the maximum payload size of the "data" element cannot exceed 4KB. This validation step takes all notification elements into account that are rendered later under the GCM "data" element, and estimates the total payload size. To maintain backward compatibility with older mobile platform message formats, the maximum allowed payload size is actually only about one third of the 4KB. See *Special Notification Elements*, below.

If any validation step fails because an invalid value was sent, the returned HTTP status is 400. The push response provides detailed information, such as the push status, which is "3" denoting an input validation error; and the push status message, which indicates the element that has failed, and why.

Example response:

```

HTTP/1.1 400 Bad Request
X-SMP-LOG-CORRELATION-ID: b1922884-4612-47a5-8654-fc01792d0bf4
Content-Type: application/JSON
{
  "status": {
    "value": "GCM notification validation error: invalid time_to_live value: 2419201",
    "code": 3
  }
}

```

These new validation steps also apply if the feature mentioned in *JSON Payload Support for GCM Push Notifications* is inactive.

Send-to-Sync

By introducing more granular validation functionality, the mobile platform GCM notification handler now supports GCM's "send-to-sync" functionality, which means the smallest possible payload can now be sent to GCM.

Example of a send-to-sync request payload:

```
{ }
```

The message sent to GCM is empty, except for the receiver's registration_id / GCM token, in a format similar to:

```
{ "to" : "bk3RNwTe3H0:CI2k_HHwgIpoDKCIZvvDMExUdFQ3P1..." }
```

Special Notification Elements

This describes payload handling for several special notification elements, and provides examples.

alert

Since a GCM message does not contain an element that corresponds to the mobile platform "alert" element, the value supplied with this field is rendered under the GCM element "data". If the GCM JSON payload feature is inactive, the value is still rendered as the URL parameter "data.alert", but is transparent to the client. Keep in mind that "alert" does not equal the GCM message "title" element, which enables backward compatibility for clients who expect the alert to be part of the "data" payload.

badge

The mobile platform notification element "badge" also has no corresponding GCM message element, and is rendered under the "data" element.

data

The mobile platform notification element "data" supports these content types:

- String - a normal string of characters
- Serialized JSON - a serialized JSON object

Previous versions of the mobile platform notification handler always sent the value as a string in the URL parameter "data.data". Now the notification handler tries to deserialize a JSON object structure from the parameter's content, and, if it succeeds, the JSON object structure renders under the GCM message element "data". To remain backward compatible, the value from "data" is also rendered under the GCM message element "data" using the key "data". This way the receiving client then can process the message as needed by accessing the deserialized JSON object, or accessing the serialized JSON string or plain text string. See the example below, which shows how the message is sent to GCM, and how the payload might be passed forward to the client.

customParameters

The mobile platform notification element "customParameters" also allows you to specify values for "alert", "badge" and "data", which, if set, take priority over their corresponding notification element fields. Aside

from this, the values are handled accordingly, and as described above, except the data element, which is additionally rendered under the GCM message element "data" and the key "cdata" as well. This approach provides backward compatibility, but reduces the maximum supported payload size, since information is duplicated.

Example Requests

The following examples show requests using the data and customParameters elements.

data element

Example request using the data element:

```
{
  "alert": "Update Available",
  "badge": 1,
  "data": "{ \"version\": \"1.13\", \"details\": { \"size\": \"14MB\" } }"
}
```

The resulting GCM message payload:

```
{
  "to": "GCMTOKEN-1234567",
  "data": {
    "version": "1.13",
    "details": {
      "size": "14MB"
    },
    "data": "{ \"version\": \"1.13\", \"details\": { \"size\": \"14MB\" } }",
    "badge": "1",
    "alert": "Update Available"
  }
}
```

customParameters element

Example request using the customParameters element:

```
{
  "customParameters": {
    "gcm.alert": "Update Available",
    "gcm.badge": 1,
    "gcm.data": "{ \"version\": \"1.13\", \"details\": { \"size\": \"14MB\" } }"
  }
}
```

The resulting GCM message payload:

```
{
  "to": "GCMTOKEN-1234567",
  "data": {
    "version": "1.13",
    "details": {
      "size": "14MB"
    },
    "data": "{ \"version\": \"1.13\", \"details\": { \"size\": \"14MB\" } }",
    "cdata": "{ \"version\": \"1.13\", \"details\": { \"size\": \"14MB\" } }",
    "badge": "1",
    "alert": "Update Available"
  }
}
```


The above examples show how "data" information is duplicated to remain backward compatible with older client applications, which expect data to be sent as string values under certain key names (such as "data", or "cdata"). The examples also show the proper GCM approach for sending a JSON payload, which is to deserialize the given "data" value into a JSON object structure.

Resend Messages

If processing on the GCM side fails, you can automatically resend messages. Resend is scheduled when the GCM response contains an error code (currently only HTTP 400 error response codes), and includes the "Retry-After" HTTP header.

If these conditions occur, the original message persists in the database, and is picked up by the push dispatcher, which runs periodically. The previously extracted value from the "Retry-After" header is honored accordingly. Configure the maximum number of retries using the parameter "push.config.async_queue_max_retries". The default value is 3.

The current implementation sends only HTTP 400 error response codes. This does not comply with the Google developer's documentation, which proposes using the "Retry-After" mechanism for 5<xx> or 200+ error response codes. To comply, you must provision the 400 error code to handle the Resend mechanism for GCM Push API feature.

Logging and Customer Events

SAP Mobile Services make an event log entry for every GCM error code returned from the GCM server. For details about the expected errors that can be returned by GCM.

Validation errors lead to a customer event log entry detailing the element for which validation failed, and why.

1.6.1.2.8.2.3 WNS Push Notification Payload Handling

WNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

Payload Handling for WNS

This section provides examples of JSON payload handling for WNS custom values.

wnsNotification

The wnsNotification entity provides payload handling for Windows. wnsNotification is validated with a higher priority than global configured properties (such as Data, Sound, Alert). The root element of the new entity, and all its direct subelements are optional.

badge:

The badge element represents a certain WNS push schema (wns/badge).

Element handling:

- Overrides the global badge configuration, whether it is a number or a string.
- The badge is sent as a separate request to the device.
- If a number, 0 clears the badge; values from 1-99 are shown as given; and any value greater than 99 are shown as 99+.
- If a string, the badge is shown as a predefined Windows glyph.

See also: <https://msdn.microsoft.com/en-us/library/windows/apps/br212849.aspx> .

rawData

The rawData element represents a certain WNS push schema (wns/raw).

Element handling:

- Overrides the global data configuration.
- Any string can be configured as rawData, and sent to the device.

version

The version element sets the version property at the notification requests (tile, toast and badge).

Element handling:

- Any string configured as version is sent to the device.
- The default is '1.0'.

lang

The lang element sets the lang property for the notification requests (tile, and toast).

Element handling: defines the language property of the content.

schema

The schema element defines which notification schemas should be sent.

Element handling:

- A list containing up to four elements.
- Restricted to one of 'BADGE', 'TILE', 'TOAST' or 'RAW'.
- The default sends all (in case content exists).

tileTemplate

The tileTemplate element sets the template property for tile schema at the binding attribute. Windows provides a selection of several predefined templates, which affect the final layout of the displayed notification at the device. The elements that can be selected depend on the device operating system version.

Element handling:

- Any string.
- Default value depends on the push content:
 - Text only: TileSquareText04
 - One picture and text: TileSquarePeekImageAndText04
 - Few pictures and text: TileWidePeekImageCollection04
 - One picture only: TileWideImage
 - Few pictures only: TileWideImageCollection

See also: <https://msdn.microsoft.com/en-us/library/windows/apps/windows.ui.notifications.tiletemplatetype.aspx> .

toastTemplate

The toastTemplate element sets the template property for toast schema at the binding attribute. Windows provides a selection of several predefined templates, which affect the final layout of the displayed notification at the device. The elements that can be selected depend on the device operating system version.

Element handling:

- Any string.
- Default value depends on the push content:
 - Text only: ToastText01
 - One picture and text: ToastImageAndText04
 - Few pictures and text: ToastImageAndText04
 - One picture only: ToastImageAndText04
 - Few pictures only: ToastImageAndText04

See also: <https://msdn.microsoft.com/en-us/library/windows/apps/windows.ui.notifications.toasttemplatetype.aspx> .

message

The message element defines the text attribute for tile and toast notifications.

Element handling:

- No, one, or more messages can be added.
- Any string.

baseUri

The baseUri element defines the baseUri property at the binding attribute for tile and toast. This property defines a central base URL which is used for all images.

Element handling: any string.

audio

The audio element defines the audio attribute for toast notifications.

Element handling: overrides the global sound configuration.

Properties:

- loop: [Boolean, optional] – defines whether the sound should be played in a loop.
- silent: [Boolean, optional] – defines whether the sound should be muted. If true, no sound is sent.
- src: [String, required] – defines the source of the sound file.

image

The image element defines the image attribute for tile and toast notifications.

Element handling: no, one, or more images can be added.

Properties:

- alt: [String, optional] – defines an alternative text if the image cannot be loaded.
- addImageQuery: [Boolean, optional] – allows Windows to add a query to the image src, the default is false.
- src: [String, required] – defines the source of the image.

commands

The commands element defines the commands attribute for toast notifications. Commands correspond to available actions that the user can take.

Element handling: no, one, or more commands can be added.

Properties: scenario [String, optional] – defines the scenario (alarm or incomingCall, the default is alarm).

Aggregations: command:

- id [String, optional] – specifies one command from the system-defined command list. (such as snooze, or dismiss).
- arguments [String, optional] – an argument string that can be passed to the associated app to provide specifics about the action that it should execute in response to the user action.

Example for JSON

The following example shows payload handling using JSON.

```
{
  "wnsNotification": {
    "badge": "attention",
    "rawData": "WnsRawData",
    "version": "1.0",
    "baseUri": "http://foo.bar/base",
    "lang": "en-US",
    "schema": ["BADGE", "TILE", "TOAST", "RAW"],
    "tileTemplate": "TileWidePeekImageCollection04",
    "toastTemplate": "ToastImageAndText04",
    "message": ["message one", "message two", "message three"],
    "audio": {
      "loop": false,
      "silent": false,
      "src": "/foo/bar"
    }
  }
}
```

```

    },
    "image": [{
      "alt": "alternative text",
      "addImageQuery": true,
      "src": "/foo/de"
    }, {
      "alt": "alternative text2",
      "addImageQuery": false,
      "src": "/foo/de"
    }, {
      "src": "/foo/de"
    }
  ],
  "commands": {
    "scenario": "alarm",
    "command": [{
      "id": "snooze",
      "arguments": "some arguments"
    }, {
      "id": "dismiss"
    }]
  }
}

```

1.6.1.2.8.3 Notification Data Sent Through HTTP Headers

Notification data can be sent by the back end as generic HTTP headers or as device platform-specific HTTP headers.

Request

URL: `https://<mobile services host>/Notification/<registration ID>`

Note

- SAP recommends that you use the URLs provided in *Notification Data Sent Using Push API*.
- Applications built on platform and later should adopt the header format `x-SMP-xxx`. To maintain backward compatibility, applications built in earlier versions can continue to use the header format `x-SUP-xxx`. However, `x-SUP-xxx` headers will be removed in future releases.

- **Generic header**

The generic HTTP header is used in the HTTP request to send any notification type such as APNs, GCM, BlackBerry, or WNS.

Header format for notification data in platform and later:

```
<X-SMP-DATA>
```

- **APNs-specific headers**

Use these APNs-specific HTTP headers to send APNs notifications via SAP Mobile Services:

Header Structure (SAP Mobile Services and later)	Consists of
<X-SMP-APNS-ALERT>	A JSON document. You can use this header or other individual headers listed in this table.
<X-SMP-APNS-ALERT-BODY>	Text of the alert message.
<X-SMP-APNS-ALERT-ACTION-LOC-KEY>	If a string is specified, this header shows an alert with two buttons: Close and View . iOS uses the string as a key to get a localized string for the correct button title instead of View . If the value is null, the system shows an alert. Click OK to dismiss the alert.
<X-SMP-APNS-ALERT-LOC-KEY>	Key to an alert-message string in a <code>Localizable.strings</code> file for the current localization.
<X-SMP-APNS-ALERT-LOC-ARGS>	Variable string values to appear in place of the format specifiers in <code>loc-key</code> .
<X-SMP-APNS-ALERT-LAUNCH-IMAGE>	File name of an image file in the application bundle. It may include the extension. Used as the launch image when you tap the action button or move the action slider. If this property is not specified, the system uses on of the following: <ul style="list-style-type: none"> • The previous snapshot • The image identified by the <code>UILaunchImageFile</code> key in the <code>Info.plist</code> file of the application • The <code>Default.png</code>.
<X-SMP-APNS-BADGE>	Number that appears as the badge on the application icon.
<X-SMP-APNS-SOUND>	Name of the sound file in the application bundle.
<X-SMP-APNS-DATA>	Custom payload data values. These values must use the JSON-structured and primitive types, such as dictionary (object), array, string, number, and Boolean.
<X-SMP-APNS-CATEGORY>	Identifies a category that is used to specify notification actions, such as contacts, or messages.
<X-SMP-APNS-CONTENT-AVAILABLE>	Indicates whether the user can take action.

- **GCM-specific headers**

Use these GCM-specific HTTP headers to send GCM notifications:

Header Structure (SAP Mobile Services and later)	Consists of
<X-SMP-GCM-COLLAPSEKEY >	An arbitrary string (such as "Updates Available") that collapses a group of like messages when the device is offline, so that only the last message is sent to the client.
<X-SMP-GCM-DATA>	Payload data, expressed as parameters prefixed with <code>data</code> and suffixed as the key.

Header Structure (SAP Mobile Services and later)	Consists of
<X-SMP-GCM-DELAYWHILEIDLE>	(Optional) Represented as 1 or true for true, any other value for false, which is the default value.
<X-SMP-GCM-TIMETOLIVE>	Number of seconds that the message remains available on GCM storage if the device is offline.

1.6.1.2.8.4 SAP Gateway Notification Support

There are no specific handling requirements for sending notifications on the SAP Gateway side. Mobile Services sends notifications using gateway-specific headers.

The SAP Mobile Services identifies the device type, and based on the device type, converts the gateway notification headers into the third-party notification context data for APNs, GCM or BES/BIS,WNS, and MPNS.

Note

Non-SAP Gateway back ends also use the headers listed below to send generic notifications; the back ends are unaware of the device platform.

SAP Gateway-specific headers that are handled by the SAP Gateway Notification Support for sending notifications

Structure Header	Consists of
x-sap-poke-title	Text of the alert message.
x-sap-poke-entriesofinterest	Number that appears as the badge on the application icon.
x-sap-poke-data	Custom payload data values. These values must use the JSON structured and primitive types such as dictionary (object), array, string, number, and Boolean.

- **APNs**

SAP Mobile Services converts the gateway notification headers into APNs notifications:

Structure Header	Consists of
x-sap-poke-title	Text of the alert message.
x-sap-poke-entriesofinterest	Number that appears as the badge on the application icon.
x-sap-poke-data	Custom payload data values. These values must use the JSON structured and primitive types such as dictionary (object), array, string, number, and Boolean.
x-sap-poke-category	Identifies a category that is used to specify notification actions, such as contacts, or messages.
x-sap-poke-sound	Identifies the sound file to be played along with a push message when using the gateway headers.

Structure Header	Consists of
x-sap-poke-content-available	Indicates that the user can take action.

- **GCM**

SAP Mobile Services converts the gateway notification headers into GCM notifications:

Header Structure	Consists of
x-sap-poke-title	An arbitrary string (such as "Updates Available") collapses a group of like messages when the device is offline, so that only the last message is sent to the client.
x-sap-poke-data	Payload data. Size should not exceed 4KB.

- **BIS/BES**

SAP Mobile Services converts the gateway notification headers into BIS/BES notifications:

Structure Header	Consists of
x-sap-poke-data	BES/BIS notification data

- **WNS**

SAP Mobile Services converts the gateway notification headers into WNS notifications:

Structure Header	Consists of
x-sap-poke-title	Text of the alert message to be shown on the Tile and Toast notifications
x-sap-poke-entriesofinterest	Number that appears as the badge on the application icon
x-sap-poke-data	Custom payload data to be sent to the device as a raw notification

1.6.1.2.8.5 Notification Sent in URL Format

Notification data can also be sent by using the REST client, using URL arguments as part of the mobile platform push endpoint, or as the delivery address URL.

Request

URL: `http[s]://<host:port>/Notification/<application connection ID>?alert=<alert>&badge=<badge>&sound=<sound>&data=<data in text format>&category=<category_name>&content_available<true/false>`

URL: `https://<mobile services host>/Notification/<application connection ID>?alert=<alert>&badge=<badge>&sound=<sound>&data=<data in text format>&category=<category_name>&content_available<true/false>`

All URL arguments (zero to many) are optional. The arguments are converted into device-type specific notifications as explained:

- **APNs**

Parameters	Description
alert	Text of the alert message.
badge	Number that appears as the badge on the application icon.
sound	Name of the sound file in application bundle.
data	Custom payload data values. These values must use the JSON-structured and primitive types, such as dictionary (object), array, string, number, and Boolean.
category	Identifies a category that is used to specify notification actions, such as contacts, or messages.
content_available	Indicates whether the user can take action.

- **GCM**

Parameters	Description
alert	An arbitrary string (such as "Updates Available") that collapses a group of like messages when the device is offline, so that only the last message is sent to the client
data	Payload data, expressed as parameters prefixed with data and suffixed as the key

- **BIS/BES**

Parameters	Description
data	Notification data
alert	Text of the alert message
badge	Number that appears as the badge on the application icon

- **WNS**

Parameters	Description
alert	The text of the alert message to be sent as a Tile notification
badge	Number that appears as the badge on the application icon
data	Payload data to be sent

- **MPNS (Notification for Windows Phone)**

Parameters	Description
alert	The text of the alert message to be sent as a Tile notification
badge	Number that appears as the badge on the application icon
data	Payload data to be sent

Based on the data send either in headers or in the URL, corresponding notification is sent to the device:

Header Notification	Tile Notification	Toast Notification	Raw Notification
Alert	Yes	Yes	No
Badge	Yes	No	No
Data	No	No	Yes

Related Information

[Native Push Notification for a Back End \[page 354\]](#)

1.6.1.2.9 Registering Clients for Native Push Notifications

Enable native push notifications, and register your application to receive push notifications.

Prerequisites

- Configure the registration ID.
- Configure the application to send push notifications.

[Registering Android Clients \[page 387\]](#)

Register and enable your Android device clients to receive push notifications.

[Registering BlackBerry Clients \[page 388\]](#)

Register and enable your BlackBerry device clients to receive push notifications.

[Registering iOS Clients \[page 389\]](#)

Register and enable your iOS device clients to receive push notifications.

[Registering Windows 8 Desktop and Tablet Clients \[page 390\]](#)

Register and enable your Windows 8 (desktop and tablet) devices to receive push notifications.

[Registering Windows 8 Phone Clients \[page 390\]](#)

Register and enable your Windows 8 Phone 8 client to receive push notifications.

1.6.1.2.9.1 Registering Android Clients

Register and enable your Android device clients to receive push notifications.

Prerequisites

- (Administrator) In the cockpit, configure the application for push notification by specifying the sender ID and API key.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

1. If `AndroidGcmPushEnabled` is enabled, the sender ID is sent in the response. Upon successful client onboarding, the response indicates the GCM push is enabled.
2. If GCM is enabled and the sender ID is available, the client uses that sender ID to register itself with GCM and get its unique GCM registration ID.
3. Specify the POST method in the URL, and include the registration ID:

```
http://<host:port>/odata/applications/{<service version>}/{appid}/Connections/  
( '{appid}' )  
Method : POST  
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" =  
"MERGE"  
Body:  
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://  
schemas.microsoft.com/ado/2007/08/dataservices/metadata" xmlns:d="http://  
schemas.microsoft.com/ado/2007/08/dataservices">  
  <content type="application/xml">  
    <m:properties>  
      <d:AndroidGcmRegistrationId>{GCM registration ID}</  
d:AndroidGcmRegistrationId>  
    </m:properties>  
  </content>  
</entry>
```

1.6.1.2.9.2 Registering BlackBerry Clients

Register and enable your BlackBerry device clients to receive push notifications.

Prerequisites

- To configure push notifications for BIS, import a BIS certificate into the `smp_keystore.jks` and `keystore` files in the `server configuration` folder.
- (Administrator) In the cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

Enable push notifications in the application:

- a. Update the application connection settings with the BES/BIS registration ID.
- a. In your application, set the `BlackberryPushListenerPort` and `BlackberryDevicePin` properties.

```
http://<host:port>/odata/applications/{<service version>}/{appid}/Connections/
('{appid}')
Method : POST
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" =
"MERGE"
Body: Http payload to update the blackberry (BES) device PIN and push port
<?xml version="1.0" encoding="utf-8"?>
  <entry xmlns="http://www.w3.org/2005/Atom" xmlns:d="http://
schemas.microsoft.com/ado/2007/08/dataservices" xmlns:m="http://
schemas.microsoft.com/ado/2007/08/dataservices/metadata">
    <m:properties>
      <d:BlackberryDevicePin> </d:BlackberryDevicePin>
      <d:BlackberryBESListenerPort><XXXX></d:BlackberryBESListenerPort>
    </m:properties>
  </content>
</entry>
Body: Http payload to update the blackberry (BIS) device PIN and push port:
<?xml version="1.0" encoding="utf-8"?>
  <entry xmlns="http://www.w3.org/2005/Atom" xmlns:d="http://
schemas.microsoft.com/ado/2007/08/dataservices" xmlns:m="http://
schemas.microsoft.com/ado/2007/08/dataservices/metadata">
    <m:properties>
      <d:BlackberryPushEnabled>true</d:BlackberryPushEnabled>
      <d:BlackberryDevicePin> </d:BlackberryDevicePin>
      <d:BlackberryPushAppID></d:BlackberryPushAppID>
      <d:BlackberryPushBaseURL> </d:BlackberryPushBaseURL>
      <d:BlackberryPushListenerPort><XXXX></d:BlackberryPushListenerPort>
    </m:properties>
  </content>
</entry>
```

1.6.1.2.9.3 Registering iOS Clients

Register and enable your iOS device clients to receive push notifications.

Prerequisites

- (Administrator) In the cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.

Procedure

Enable push notifications in the application:

- To receive the device token, implement the `application:didRegisterForRemoteNotificationsWithDeviceToken` method in your application delegate.
- Update the `ApnsDeviceToken` and `DeviceType` properties via a POST request. The HTTP header must include the `X-SMP-APPCID` and `Authorization` headers.

```
https://<host:port>/odata/applications/{<service version>}/{appid}/
Connections/('{appid}')
Method : POST
HTTP Headers : "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" =
"MERGE"
Body:
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://
schemas.microsoft.com/ado/2007/08/dataservices/metadata"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xml:base="http://host:port/odata/applications/v1/com.example.IOS/">
  <id>https://{application URL}/odata/applications/{<service version>}/
e2eTest/Connections('32552613-470f-45e0-8acc-b7d73d501682')</id>
  <content type="application/xml">
    <m:properties>
      <d:ApnsDeviceToken>{APNS device token received by the application from
APNS}</d:ApnsDeviceToken>
      <d:DeviceType>iPhone</d:DeviceType>
    </m:properties>
  </content>
</entry>
```


1.6.1.2.9.4 Registering Windows 8 Desktop and Tablet Clients

Register and enable your Windows 8 (desktop and tablet) devices to receive push notifications.

Prerequisites

- (Administrator) In Mobile Services cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

1. To obtain the channel URI, register the application with WNS. See [Push notification overview \(Windows Store apps\)](#)  on the Windows Dev Center Web site.
2. Check the value that is returned from `WnsPushEnable` during registration, and if the value is true, continue with either the WNS or notification registration processing. Set the `WnsChannelURI` value that is received from the application.
3. Update the application connection settings with the registration ID:

```
https://host:port/odata/applications/{<service version>}/{appid}/Connections/  
( '{appid}' )  
Method : POST  
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" =  
"MERGE"  
Body:  
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://  
schemas.microsoft.com/ado/2007/08/dataservices/metadata" xmlns:d="http://  
schemas.microsoft.com/ado/2007/08/dataservices">  
  <content type="application/xml">  
    <m:properties>  
      <d:WnsChannelURI>{WNS Channel URI}</d:WnsChannelURI>  
    </m:properties>  
  </content>  
</entry>
```

1.6.1.2.9.5 Registering Windows 8 Phone Clients

Register and enable your Windows 8 Phone 8 client to receive push notifications.

Prerequisites

- (Administrator) In Mobile Services cockpit, configure the application for push notification.

- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

1. In the cockpit, configure push notification. Specify the device type during application connection and registration, and ensure that the HTTP header includes the X-SMP-APPCID and Authorization headers.
2. To obtain the channel URI, register the application with the Microsoft Push Notification Service (MPNS). See [Push notifications for Windows Phone](#) on the Windows Phone Dev Center Web site.
3. Check the value of `MpnsPushEnable` that is returned during registration, and if the value is true, continue with either the MPNS or notification registration processing. Set the `MpnsChannelURI` value that is received from the application.
4. Using the ApplicationConnection ID (`<appid>`) that is returned from the mobile platform registration call (in either the X-SMP-APPCID HTTP header or the `ApplicationConnectionId` property), update the `MpnsChannelURI` property for the application connection using the Channel URI returned by the application:

```
https://host:port/odata/applications/{<service version>}/{appid}/Connections/
('{appid}')
Method : POST
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" =
"MERGE"
Body:
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://
schemas.microsoft.com/ado/2007/08/dataservices/metadata" xmlns:d="http://
schemas.microsoft.com/ado/2007/08/dataservices">
  <content type="application/xml">
    <m:properties>
      <d:MpnsChannelURI>{MPNS Channel URI}</d:MpnsChannelURI>
    </m:properties>
  </content>
</entry>
```

1.6.1.2.10 Service Document

Get the service document for the application connection.

Usage

Retrieving the service document allows the client to discover the capabilities and locations of the available collections.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>`

HTTP Method: *GET*

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><service version></code>	Mandatory	v1 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp HTTP/1.1
Host: mobile-ops-subscriber-development-qa-xapp8.cfapps.sap.hana.ondemand.com
Authorization: Basic REVWMDAwMTppbml0aWFs
```

Response Body Example

```
<?xml version='1.0' encoding='utf-8'?>
<service xmlns="http://www.w3.org/2007/app"
  xml:base="http://host:port/appSettings/odata/v1/appid/"
  xmlns:atom="http://www.w3.org/2005/Atom"
  xmlns:app="http://www.w3.org/2007/app">
  <workspace>
    <atom:title>Default</atom:title>
    <collection href="Connections">
      <atom:title>Connections</atom:title>
    <collection href="Endpoints">
      <atom:title>Endpoints</atom:title>
    </collection>
  </workspace>
</service>
```

Response

Code	Description
200 OK	Returns service document

1.6.1.2.11 Logging Out Users

Terminate an active user session with the user logout service. This is useful for multiple user scenarios.

The user logout service always returns HTTP status 204 (no content), even if there was no active session for the calling user at the point the call was made. The service does not require authentication, but it does require a session cookie header identifying the session to be terminated.

Related Information

[Logout Service \[page 444\]](#)

1.6.1.2.12 Storage Service

As an application developer, you can use REST services to store device application configuration data in the server.

The application configuration data is stored based on user or device preference. The storage service stores flexible data structure and supports application-level, user-level, and device-level storage. SAP Mobile Services offers authorization and authentication schemes that secure the data.

Related Information

[Storage Service API \[page 447\]](#)

1.6.1.2.13 Client Resources Service

As an application developer, you can use REST services to download resources from the server.

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Mobile Services Admin Cockpit.

1.6.1.2.14 Client Log Upload Service

As an application developer, you can use REST services to upload client logs to the server.

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

1.6.1.2.15 Push as API Service

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

Developers enable native push notification in the application code, and link the corresponding certificate with the mobile application at build time. Users download the application from a market place, such as Apple App

Store, Google Play, or similar service, and, when a change occurs in the back end, a push notification is sent to mobile applications on devices that have push enabled.

1.6.1.2.16 Client Usage Report and User Feedback Upload Service

The Client Usage Report and User Feedback Upload service allows an application to upload a client usage report and user feedback for SAP Mobile Services.

1.6.1.2.17 Role Service

As an application developer, you can use REST services to enable mobile devices to retrieve user name and roles associated with the user according to the System for Cross-domain Identity Management (SCIM) protocol. The role service allows you to get the logical roles for the current user, and you can use it to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

The service implements:

- The `/Me` authenticated subject alias, which enables the client to use a URL in the form `<base-URI>/ME` as a URI alias for the user who is associated with the currently authenticated subject for a SCIM operation. The service provider may grant the request or redirect it.
- Multi-valued attributes, which enable the client to retrieve the roles that are associated with the user.
- The `/Me` service can access attributes that pass through the user's browser, and are not secret. For example, a SAML attribute such as `"name"` or `"email"` may pass through the browser, and the application can use that information to show the user's real name instead of `"username"` and populate the email field. Name and email fields are set only if they are available in the SAML attributes and can be retrieved correctly. These values can also be available if the authentication method OAuth was selected for the mobile application.

1.6.1.3 Reference

Describes REST API resources.

[HTTP Headers and Cookies \[page 396\]](#)

Use HTTP headers and cookies to retrieve application connection information.

[HTTP Headers Used to Propagate User IDs \[page 396\]](#)

Enables the back end to use the information in the `X-SMP-ENDUSERNAME` `<username>` header to identify the user who sent a request.

[Supported Onboarding Services \[page 397\]](#)

Lists the general naming conventions of the services supported for registration and onboarding purposes.

[Proxy Responses \[page 399\]](#)

Proxy responses include all the cookies and headers from the proxied back end.

[Application Connections \[page 399\]](#)

Methods for creating, updating, or reading application connections.

[Error Codes and Message Formats \[page 417\]](#)

The server returns different formats for error codes and messages according to different "Accept" values in request headers.

[External Access to Application Versions \[page 417\]](#)

Provides third-party access to meta-data and binary information about published artifacts stored in the App Lab service. These APIs require authentication and authorization using a service-key.

[Authenticate Applications Using SAML 2.0 \[page 428\]](#)

Initiate a REST service call to create SAML 2.0 assertion for authenticating the application security configuration.

[Retrieve Customization Resource Bundles \[page 433\]](#)

Download application resource bundles.

[Accessing Services Through Proxy URLs \[page 434\]](#)

To access a back end or Internet-based service, use a proxy URL that supports read, create, update, delete, merge and patch.

[Feature Restriction Policies \[page 436\]](#)

REST API methods for managing feature restriction policies for an application. You can get, update, or remove features enabled through the Java API, `isEnabled()`. Any enabled feature can be disabled by the administrator through the cockpit, providing additional control.

[Upload Logs and Traces \[page 440\]](#)

Upload client logs and Business Transaction XML (BTX) files for analysis.

[Cross-Origin Resource Sharing Requests \[page 443\]](#)

The cross-origin resource sharing (CORS) standard includes HTTP headers that allow you to control resource access from Web browsers. CORS also defines a preflight HTTP OPTIONS method that requests a list of the supported methods for a resource; if the requested method is supported, the actual HTTP request is sent.

[Logout Service \[page 444\]](#)

The user logout service provides functionality to terminate an active user session from the client.

[Logout V2 Service \[page 446\]](#)

The user logout service provides functionality to invalidate the sessions and revoke all the refresh tokens of a user.

[Storage Service API \[page 447\]](#)

Storage service facilitates application developers to persist mobile application specific data, such as user preferences, user contextual data, and application configuration.

[Client Resources Service \[page 475\]](#)

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Mobile Services Admin Cockpit.

[Client Log Upload Service \[page 484\]](#)

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

[Push as API Service \[page 490\]](#)

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

[Bundle Service \[page 507\]](#)

The Bundle Service API allows you to manage resource bundles directly for apps.

[Offline Store Upload Service \[page 510\]](#)

The Offline Store Upload API enables a client to upload offline store (database files) to the server. If Offline Store Upload is enabled for an application, you can upload the offline store (database files) into the server. The upload process creates a zip file that includes the uploaded database files, and is then saved into the server database.

[Client Usage Report and User Feedback Upload Service \[page 517\]](#)

The Client Usage Report and User Feedback Upload service allows an application to upload client usage and user feedback charts for SAP Mobile Services.

[Role Service \[page 520\]](#)

This service allows you to get the logical roles that are assigned to the current user, which you can use to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

1.6.1.3.1 HTTP Headers and Cookies

Use HTTP headers and cookies to retrieve application connection information.

Note

In the current platform versions, applications should adopt the header format X-SMP-XXX. To maintain backward compatibility, applications built in earlier versions can continue to use the header format X-SUP-XXX. However, these headers will be removed from future releases, and you should update your applications to use the X-SMP-XXX header format. .

1.6.1.3.2 HTTP Headers Used to Propagate User IDs

Enables the back end to use the information in the X-SMP-ENDUSERNAME `<username>` header to identify the user who sent a request.

Consider a scenario in which only the technical user ID, and not the user name of the mobile device appears in the back end. An administrator can now set the `Propagate User Name` option to true in SAP Mobile Services, which adds an X-SMP-ENDUSERNAME `<username>` HTTP header to outgoing requests, and securely passes the user name to the back end. The back-end service uses the information to apply additional filters that ensure that the information about the user who sent the request appears, even though the technical user made the request.

Related Information

[Defining Connectivity \[page 89\]](#)

1.6.1.3.3 Supported Onboarding Services

Lists the general naming conventions of the services supported for registration and onboarding purposes.

Whenever there is a change in the functionality newer version of onboarding services change in order to exchange data back and forth between client and server. Supported onboarding services to handle the requests which is OData compliant are:

Onboarding Service Versions	Impact
v1	<p>Initial version of the onboarding service.</p> <p>This service deviates from the standard OData compliant service in the sense that HTTP PUT requests can be used to change the individual properties, while parameters which are not included in the request remain same. When sending the complete entity payload, it however replaces the entity as expected.</p> <p>The service also includes a PATCH service, which must be accessed by tunneling the request as a POST request with the header <i>X-HTTP-METHOD: MERGE</i>.</p> <p>Example PATCH request*:</p> <pre>POST /<someentity> X-HTTP-METHOD:MERGE { "key" : "value" }</pre> <p>This request updates the key field of the entity.</p> <div><p>Note</p><p>Furthermore, the feature vector field is modeled as a collection of complex types which is not a valid ODATA v2 construct. ODATA client libraries parsing the service metadata may report an error with v1 version of the service.</p></div>
v2	<p>Introduced changes in the semantics of the OData to ensure ODATA compliance.</p> <p>The PUT operation now updates/deletes any fields from the entity, if they are not included in the request payload. Clients must use a PATCH operation as described in the v1 service. Additionally, with v2 version, you can use the PATCH HTTP verb, instead of tunneling it with the special header in an HTTP POST request.</p> <p>Example POST request*:</p> <pre>POST /<someentity> X-HTTP-METHOD:MERGE { "key" : "value" }</pre>

Onboarding Service Versions	Impact
	<p>Or,</p> <p>Sample PATCH request:</p> <pre>PATCH /<someentity> X-HTTP-METHOD:MERGE { "key" : "value" }</pre> <p>The feature vector in v2 is now modeled as an entity and referenced in this way from other entities. The metadata of the v2 service should be parsable by ODATA client libraries". For more information, see <i>Feature Restriction Policies</i>.</p>
v3	<p>Introduced the device capabilities. For more information, see <i>Create Application Connection with Capability Handling</i>.</p>
v4	<p>Introduced a new onboarding contract for native applications only.</p> <p>This service has new application connection properties such as <code>UserName</code>, <code>UserLocale</code>, <code>TimeZone</code>, <code>LastKnownLocation</code>, <code>CreatedAt</code>, <code>PushGroup</code>, <code>Email</code>, <code>IsWiped</code>, <code>DeviceId</code>, <code>NetworkPolicy</code>, <code>LogMaxFileSize</code>, and <code>LogMaxFileNum</code>.</p> <p>During onboarding, the GET operation can now receive a read-only property for <code>UserName</code> and <code>CreatedAt</code>.</p>
latest	<p>Always refers to the latest version of the onboarding service.</p> <div data-bbox="804 1301 1390 1552"> <p>Note</p> <p>The behavior of this service could change anytime if an updated version of the onboarding service is included in the server version. It is recommended to build your clients against a fixed server URL with a constant version.</p> </div>
<div data-bbox="193 1597 1390 1682"> <p>Note</p> <p>* Indicates a sample request and not all the headers are shown here.</p> </div>	

Related Information

[Feature Restriction Policies \[page 436\]](#)

1.6.1.3.4 Proxy Responses

Proxy responses include all the cookies and headers from the proxied back end.

1.6.1.3.5 Application Connections

Methods for creating, updating, or reading application connections.

Note

Application connection service is implemented as an OData service and therefore follows OData standards.

[Metadata \[page 400\]](#)

Get the metadata document, which includes the metadata for the application connection settings and proxy endpoints.

[Retrieve Changed Settings and Connections Metadata \[page 401\]](#)

You can retrieve only the changed settings and connections metadata.

[Create Application Connection \[page 402\]](#)

Create an application connection and initially set the application connection settings.

[Create Application Connection with Capability Handling \[page 403\]](#)

Enable the client to manage form factor and capabilities in the application connection.

[Get Application Settings \[page 410\]](#)

Retrieve application settings for the application connection.

[Get Proxy Endpoints \[page 411\]](#)

Get all proxy endpoints for the application connection.

[Get Proxy Endpoint by Endpoint Name \[page 412\]](#)

Get a specific endpoint by specifying the endpoint name.

[Get Application Property Settings \[page 413\]](#)

Get the specific property value for a property from the application settings.

[Update Application Settings \[page 414\]](#)

Update the application settings with the properties in the request.

[Delete Application Connection \[page 415\]](#)

Delete an application connection.

[Revoke User Token \[page 416\]](#)

Revoke all tokens registered to a specific user for an application, from the server.

1.6.1.3.5.1 Metadata

Get the metadata document, which includes the metadata for the application connection settings and proxy endpoints.

Usage

Metadata documents are based on the OData standard and are required to implement application connection services.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/$metadata`

HTTP Method: `GET`

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><service version></code>	Mandatory	v1 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/$metadata HTTP/1.1
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic REVWMDAwMTppbmI0aWFs
```

Response

Code	Description
200 OK	Returns service document

Response Body Example

```
<?xml version="1.0" encoding="utf-8"?>
<edmx:Edmx Version="1.0" xmlns:edmx="http://schemas.microsoft.com/ado/2007/06/edmx"
  xmlns:smp="http://www.sap.com/smp/odata"><edmx:DataService
  m:DataServiceVersion="2.0" xmlns:m="http://schemas.microsoft.com/ado/2007/08/
```



```

dataservices/metadata"><Schema Namespace="applications" xmlns="http://
schemas.microsoft.com/ado/2008/09/edm">
<EntityType Name="Endpoint">
  <Key>
    <PropertyRef Name="EndpointName"></PropertyRef></Key>
  </EntityType>
<EntityType Name="Connection">
  <Key>
    <PropertyRef Name="ApplicationConnectionId"></PropertyRef></Key>
    <Property Name="ETag" Type="Edm.String" Nullable="false"
sup:ReadOnly="true"></Property>
    <Property Name="ApplicationConnectionId" Type="Edm.String" Nullable="false"
sup:ReadOnly="true"></Property>
    <Property Name="AndroidGcmPushEnabled" Type="Edm.Boolean" Nullable="false"
sup:ReadOnly="false"></Property>
    <Property Name="AndroidGcmRegistrationId" Type="Edm.String" Nullable="true"
sup:ReadOnly="false"></Property>
    <Property Name="AndroidGcmSenderId" Type="Edm.String" Nullable="true"
sup:ReadOnly="true"></Property>
  </EntityType>
<EntityContainer Name="Container" m:IsDefaultEntityContainer="true">
<EntitySet Name="Connections" EntityType="applications.Connection">
</EntitySet>
</EntityContainer>
</Schema>
</edmx:DataServices>
</edmx:Edmx>

```

1.6.1.3.5.2 Retrieve Changed Settings and Connections Metadata

You can retrieve only the changed settings and connections metadata.

To retrieve changed application settings information, issue a GET request to this URL:

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Connections('<appid>')?If-None-Match="$ {ETag} "`

HTTP Method: [GET](#)

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><service version></code>	Mandatory	v1 onwards

The `$ {ETag}` part of the URL is a version identifier that is included in the response of the GET method. If the ETag value of the current application settings is the same as the value in the request, a status code 304 without a response body is returned to the client to indicate that there are no application setting changes.

1.6.1.3.5.3 Create Application Connection

Create an application connection and initially set the application connection settings.

Usage

All application connection settings are optional, the minimal body contains no properties at all. Mobile platform populates default values as needed.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Connections`

HTTP Method: `POST`

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
		<div>Note If an application is configured for anonymous access in the cockpit, the registration is successful even if there are no credentials, or incorrect ones, in the authorization header.</div>
<code><host:port></code>	Mandatory	Host name should match the domain registered with mobile platform. If the requested domain name does not match, a default domain is used.
<code><service version></code>	Mandatory	v1 onwards

Request Body Example

```
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
<title type="text"/>
<updated>2014-061-15T02:23:29Z</updated>
<author>
<name/>
</author>
<category term="applications.Connection" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
```

```
<content type="application/xml">
<m:properties>
<d:DeviceType>iPhone</d:DeviceType>
<d:DeviceModel m:null="true" />
<d:ApnsDeviceToken
m:null="false">18AA4813FB9E6393065BFEDADCDD68173782A42599F3C9E2BF14F990F2D9F096</
d:ApnsDeviceToken>
</m:properties>
</content>
</entry>
```

Response

Code	Description
201 Created	New application connection settings are included in the response body.

Related Information

[Create Application Connection \[page 348\]](#)

[Cross-Origin Resource Sharing Requests \[page 443\]](#)

[Feature Restriction Policies \[page 436\]](#)

1.6.1.3.5.4 Create Application Connection with Capability Handling

Enable the client to manage form factor and capabilities in the application connection.

The device sends its form factor (such as smartphone or tablet), and capabilities [such as purchaseOrder-display, or a wildcard (*) in case the device has all the capabilities] during registration, or when the application connection is updated. You can request a list of capabilities from the device. When the device user adds or removes a capability, the application connection is updated.

[Create Capabilities upon Registration \[page 404\]](#)

Store the device's form factor and its capabilities when the device is registered.

[Create Capabilities upon Update \[page 406\]](#)

Update an existing application connection, using a POST method.

[List Capabilities on a Device \[page 407\]](#)

Obtain a list of all the capabilities of a device.

[Delete Capabilities \[page 409\]](#)

Delete an application capability, for example, a user may elect to remove the capability to receive an e-mail notification. The server is notified of this change to the application connection.

Related Information

[Push API Notification Scenarios \[page 357\]](#)

[Push-to-Capability Scenario \[page 367\]](#)

[Create Application Connection with Capability Handling \[page 348\]](#)

[Feature Restriction Policies \[page 436\]](#)

1.6.1.3.5.4.1 Create Capabilities upon Registration

Store the device's form factor and its capabilities when the device is registered.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appID>/Connections`

HTTP Method *POST*

Request Parameters

Parameter	Type	Description
<code><appID></code>	Mandatory	ID that uniquely identifies an application
<code><service version></code>	Mandatory	v3 onwards

Request Example

```
POST
http://localhost:8082/odata/applications/latest/
TESTAPP81dle7af59f84f78a342ead3ad2a22a4/Connections
Content-Type: application/atom+xml Authorization: Basic
UDE5NDA3MDMyNDU6U2VjcV0MTI=
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
<link
rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Capability"
type="application/atom+xml;type=feed" title="Capability">
<m:inline>
<feed>
<entry>
<content type="application/xml">
<m:properties>
<d:Category>push</d:Category>
<d:CapabilityName>purchaseOrder-display</d:CapabilityName>
</m:properties>
</content>
</entry>
</feed>
</m:inline>
```

```

</link>
<content type="application/xml">
<m:properties>
<d:FormFactor>smartphone</d:FormFactor>
<d:DeviceType>iPhone</d:DeviceType>
</m:properties>
</content>
</entry>

```

Response Example

```

HTTP/1.1 201 Created set-cookie:
X-SMP-SESSID=14DBBA390DC7598A482B567006E109332628AB966D86E3867553E86092957BF9;
Path=/; HttpOnly set-cookie:
JTENANTSESSIONID_hmtenant1=1Xurlcqv8KCcWVylL1T5UflsgqmyTg1L35zLjTnmEps%3D;
Path=/; HttpOnly set-cookie:
X-SUP-APPCID=bd985307-497d-417e-b6fc-08f089d503f7; Expires=Wed,
11-Jul-2035 10:57:16 GMT; Path=/ set-cookie:
X-SMP-APPCID=bd985307-497d-417e-b6fc-08f089d503f7; Expires=Wed,
11-Jul-2035 10:57:16 GMT; Path=/ dataserviceversion: 1.0 date: Thu, 16
Jul 2015 10:57:16 GMT location:
http://localhost:8082/odata/applications/latest/
TESTAPP81dle7af59f84f78a342ead3ad2a22a4/Connections('bd985307-497d-417e-
b6fc-08f089d503f7')
content-type: application/atom+xml;charset=utf-8 server: SAP
<entry
xml:base="http://localhost:8082/odata/applications/latest/
TESTAPP81dle7af59f84f78a342ead3ad2a22a4/"
xmlns="http://www.w3.org/2005/Atom"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">
<id>http://localhost:8082/odata/applications/latest/
TESTAPP81dle7af59f84f78a342ead3ad2a22a4/Connections('bd985307-497d-417e-
b6fc-08f089d503f7')</id>
<title type="text"/>
<updated>2015-07-16T10:57:16Z</updated>
<author>
<name/>
</author>
<link rel="edit" title="Connection" href="Connections('bd985307-497d-417e-
b6fc-08f089d503f7')"/>
<link
rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Capability"
type="application/atom+xml;type=feed" title="Capability"
href="Connections('bd985307-497d-417e-b6fc-08f089d503f7')/Capability">
<m:inline/>
</link>
<link
rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
FeatureVectorPolicy"
type="application/atom+xml;type=feed" title="FeatureVectorPolicy"
href="Connections('bd985307-497d-417e-b6fc-08f089d503f7')/FeatureVectorPolicy">
<m:inline/>
</link>
<category term="applications.Connection"
scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
<content type="application/xml">
<m:properties>
<!-- [...] -->
<d:DeviceType>iPhone</d:DeviceType>
<!-- [...] -->
<d:FormFactor>smartphone</d:FormFactor>
</m:properties>
</content>
</entry>

```

Response

Code	Description
201 OK	The application connection has been created.

1.6.1.3.5.4.2 Create Capabilities upon Update

Update an existing application connection, using a POST method.

Usage

Update the form factor and capabilities when they are changed on the device (update `app.connection`).

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appId>/Connections(<registrationId>)`

HTTP Method *POST*

Request Example

```
POST https://
localhost:8082/odata/applications/latest/TESTAPPedb7f2619b66474ea5ae6376abdb03b2/
Connections('f12aa913-6726-4ab8-ac77-e309325b86a7') HTTP/1.1
Accept-Encoding: gzip,deflate
Content-Type: application/xml
Authorization: Basic UDIwMDIzMzY5MzY6JW5CaTdCIzBTaz9VaGZ6UnA=
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
<link
rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/Capability"
type="application/atom+xml;type=feed" title="Capability">
<m:inline>
<feed>
<entry>
<content type="application/xml">
<m:properties>
<d:Category>push</d:Category>
<d:CapabilityName>capability_on_update</d:CapabilityName>
<d:CapabilityValue>example</d:CapabilityValue>
</m:properties>
</content>
</entry>
</feed>
</m:inline>
</link>
```

```
<content type="application/xml">
<m:properties>
<d:DeviceType>Android</d:DeviceType>
<d:FormFactor>phone</d:FormFactor>
</m:properties>
</content>
</entry>
```

Response Example

```
HTTP/1.1 200 OK
X-SMP-CLUSTER-MEMBER: b01126135f554e7c96e4d01cc515b013
X-SMP-LOG-CORRELATION-ID: 44d40b47-9a7c-4b47-9324-53a90740944e
Set-Cookie: X-SMP-
SESSID=DED50CE91DA480F31DD661F209A189FEECA3B1B0328A5E8BEDC0C856834F1291; Path=/;
Secure; HttpOnly; SameSite=None
Set-Cookie:
JTENANTSESSIONID_tnv2kh5r33=1o4vIevWUIvOZd1ABBVrSeimVf0zYm%2F%2F18i5CAEKIA%3D;
Domain=.hana.ondemand.com; Path=/; Secure; HttpOnly; SameSite=None
Cache-Control: no-store, no-cache, max-age=0
Set-Cookie: X-SUP-APPCID=f12aa913-6726-4ab8-ac77-e309325b86a7; Expires=Thu, 31-
May-2040 06:36:49 GMT; Path=/; Secure; HttpOnly
Set-Cookie: X-SMP-APPCID=f12aa913-6726-4ab8-ac77-e309325b86a7; Expires=Thu, 31-
May-2040 06:36:49 GMT; Path=/; Secure; HttpOnly
DataServiceVersion: 1.0
Date: Fri, 05 Jun 2020 06:36:49 GMT
Content-Type: text/xml
Content-Length: 0
Server: SAP
Set-Cookie: BIGipServerdevautomationhanamobile.int.sap.eu2.hana.ondemand.com=!
Pe9JMz+lsxeoIgjsonnpQvDoyMkQcDuZOIf9aecnbObeuxpkJGyY89+pyAXG1GOvpyHAFMwOB87PPgg=;
path=/; Httponly; Secure; SameSite=None
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload
```

Response

Code	Description
200 OK	The existing application connection has been updated.

1.6.1.3.5.4.3 List Capabilities on a Device

Obtain a list of all the capabilities of a device.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appId>/Connections(<registrationId>)/Capability`

HTTP Method [GET](#)

Request Parameters

Parameter	Type	Description
<appId>	Mandatory	ID that uniquely identifies an application
<registrationId>	Mandatory	The connection ID of the application instance that is interacting with the service
<service version>	Mandatory	v3 onwards

Request Example

```
GET
http://localhost:8082/odata/applications/latest/
TESTAPPd890b4182e5b42238c89e743e4196731/Connections('350a85d5-916c-4442-81b6-a2192481cdcd')/Capability
HTTP/1.1 Authorization: Basic UDE5NDA3MDMyNDU6U2VjcmV0MTI=
X-SMP-APPCID: 350a85d5-916c-4442-81b6-a2192481cdcd
```

Response Example

```
<feed
  xml:base="http://localhost:8082/odata/applications/latest/
TESTAPPd890b4182e5b42238c89e743e4196731/"
  xmlns="http://www.w3.org/2005/Atom"
  xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
  xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">
  <title type="text">Capabilities</title>
  <id>http://
localhost:8082/odata/applications/latest/TESTAPPd890b4182e5b42238c89e743e4196731/
Connections('350a85d5-916c-4442-81b6-a2192481cdcd')/Capability</id>
  <updated>2015-07-16T13:16:51Z</updated>
  <link rel="self" title="Capabilities" href="Capabilities"/>
  <entry>
  <id>http://
localhost:8082/odata/applications/latest/TESTAPPd890b4182e5b42238c89e743e4196731/
Capabilities(ApplicationConnectionId='350a85d5-916c-4442-81b6-a2192481cdcd',
CapabilityName='purchaseOrder-display',Category='push')</id>
  <title type="text"/>
  <updated>2015-07-16T13:16:51Z</updated>
  <author>
  <name/>
  </author>
  <link rel="edit" title="Capability"
href="Capabilities(ApplicationConnectionId='350a85d5-916c-4442-81b6-a2192481cdcd',
CapabilityName='purchaseOrder-display',Category='push')"/>
  <category term="applications.Capability" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <content type="application/xml">
  <m:properties>
  <d:Category>push</d:Category>
  <d:CapabilityName>purchaseOrder-display</d:CapabilityName>
  <d:ApplicationConnectionId>350a85d5-916c-4442-81b6-a2192481cdcd</
d:ApplicationConnectionId>
  <d:CapabilityValue>example</d:CapabilityValue>
  </m:properties>
  </content>
  </entry>
</feed>
```


Response

Code	Description
200 OK	The list of capabilities has been created.

1.6.1.3.5.4.4 Delete Capabilities

Delete an application capability, for example, a user may elect to remove the capability to receive an e-mail notification. The server is notified of this change to the application connection.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appId>/Capabilities(ApplicationConnectionId='<registrationID>',CapabilityName='<capability_name>',Category='<capability_category>')`

HTTP Method *DELETE*

Request Parameters

Parameter	Type	Description
<code><appId></code>	Mandatory	ID that uniquely identifies an application
<code>registrationId</code>	Mandatory	The connection ID of the application instance that is interacting with the service
<code><capability_name></code>	Mandatory	The capability name used in the application, such as "purchaseOrder-display".
<code><capability_category></code>	Mandatory	The capability category, such as "push".
<code><service version></code>	Mandatory	v3 onwards

Request Example

```
DELETE http://
localhost:8082/odata/applications/latest/TESTAPPea4a626ffccc4ef98dbf27da343132aa/
Capabilities(ApplicationConnectionId='784b1fb8-4da5-4c87-
a2b5-282bb4506253',CapabilityName='purchaseOrder-display',Category='push')
Authorization: Basic UDE5NDA3MDMyNDU6U2VjcV0MTI= X-SMP-APPCID:
784b1fb8-4da5-4c87-a2b5-282bb4506253
```

Response

Code	Description
200 OK	The capability was removed from the application connection.

1.6.1.3.5.5 Get Application Settings

Retrieve application settings for the application connection.

Usage

You can retrieve application settings by either explicitly specifying the application connection ID, or by having the application connection ID determined from the call context (that is, from either the X-SMP-APPCID cookie or X-SMP-APPCID HTTP header, if specified). On the first call, you can simplify your client application code by having the application connection ID determined from the call context.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Connections('&<appid>')`

HTTP Method: *GET*

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><appid></code>	Mandatory	The connection ID of the application instance interacting with the service
<code><service version></code>	Mandatory	v1 onwards
<code><user_name></code>	Optional (read-only)	v4 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-bcaa-439d-9741-660a3cb56771') HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic <XXXX>
```

Response

Code	Description
200 OK	Returns service document

Response Body Example

```
<?xml version="1.0" encoding="UTF-8"?>
-<entry xml:base="http://10.53.138.170:8080/odata/applications/latest/G3T/"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
xmlns="http://www.w3.org/2005/Atom">
<id>http://10.53.138.170:8080/odata/applications/latest/G3T/
Connections<XXXX></id>
<title type="text"/>
<category scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"
term="applications.Connection"/>
<content type="application/xml">
<m:properties>
  <d:Etag>2013-11-07 14:44:43.0</d:Etag><d:ApplicationConnectionId>xxxx</
d:ApplicationConnectionId>
  <d:AndroidGcmPushEnabled m:type="Edm.Boolean">false</d:AndroidGcmPushEnabled>
  <d:AndroidGcmRegistrationId m:null="true"/><d:AndroidGcmSenderId/>
  <d:ApnsPushEnable m:type="Edm.Boolean">true</d:ApnsPushEnable>
  <d:ApnsDeviceToken m:null="true"/>
  <d:MpnsPushEnable m:type="Edm.Boolean">true</d:MpnsPushEnable>
  <d:ProxyApplicationEndpoint>http://vmw3815.wdf.sap.corp:50009/sap/opu/sdata/
iwfnd/RMTSAMPLEFLIGHT/</d:ProxyApplicationEndpoint>
  <d:ProxyPushEndpoint>http://INLC50802847A:8080/Notification</d:ProxyPushEndpoint>
  <d:UploadLogs>false</d:UploadLogs>
  <d:WnsChannelURI m:null="true"/>
  <d:WnsPushEnable m:type="Edm.Boolean">false</d:WnsPushEnable>
</m:properties>
</content>
</entry>
```

1.6.1.3.5.6 Get Proxy Endpoints

Get all proxy endpoints for the application connection.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Endpoints`

HTTP Method: *GET*

Request Parameters

Parameter	Type	Description
<appid>	Mandatory	ID that uniquely identifies an application
<service version>	Mandatory	v1 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Endpoints HTTP/1.1
Host: smpserver:8080
X-SMP-APPCID=9dffe5e9-5768-47a6-8220-144a2e0c751d
```

Response

Code	Description
200 OK	Returns service document

1.6.1.3.5.7 Get Proxy Endpoint by Endpoint Name

Get a specific endpoint by specifying the endpoint name.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Endpoints('<endpoint>')`

HTTP Method: `GET`

Request Parameters

Parameter	Type	Description
<appid>	Mandatory	ID that uniquely identifies an application
<endpoint>	Mandatory	The proxy endpoint name
<service version>	Mandatory	v1 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Endpoints('endpoint1') HTTP/1.1
Host: smpserver:8080
X-SMP-APPCID=9dffe5e9-5768-47a6-8220-144a2e0c751d
```

Response

Code	Description
404 - not found	Client tries to retrieve an endpoint that does not exist
400 - bad request	Client tries to fetch invalid property name
200 OK	OData response for endpoint-related information, which contains a remote URL and endpoint names and verifies whether anonymous access is allowed or not

1.6.1.3.5.8 Get Application Property Settings

Get the specific property value for a property from the application settings.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Connections('<registrationID>')/<property-name>`

HTTP Method: *GET*

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><registrationID></code>	Mandatory	The registration ID of the application instance that is interacting with the service
<code><property-name></code>	Mandatory	The property name can be appended to the URL to retrieve the value of a specific property
<code><service version></code>	Mandatory	v1 onwards

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-bcaa-439d-9741-660a3cb56771')/DeviceType HTTP/1.1
Cookie: X-SMP-APPCID=b6d50e93-bcaa-439d-9741-660a3cb56771; X-SMP-SESSID=97ts80gwhxkc
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic REVWMDAwMTppbml0aWFs
```

Response

Code	Description
200 OK	Returns service document

1.6.1.3.5.9 Update Application Settings

Update the application settings with the properties in the request.

Usage Information

Condition	For Service Versions	Use HTTP Operation	Properties
To replace an entity	v1 onwards	PUT	Add complete entity in the payload
To patch individual property within an entity	v1 onwards	POST	Add header X-HTTP-METHOD: MERGE and enter the properties to patch in the payload
	v1 onwards	PATCH	Enter the properties to patch in the payload

Request

URL: `https://<mobile services host>/[public/]odata/applications/<service version>/<appid>/Connections('&<registrationID>':<Version>]`

HTTP Method: `PUT`

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	The application ID that uniquely identifies the application
<code><registrationID></code>	Mandatory	The registrationID of the application instance that is interacting with the service
<code><service version></code>	Mandatory	v1 onwards

Request Header Example

```
PUT /odata/applications/v1/com.sap.myapp/Connections('<XXXX>') HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Content-Length: 4744
Content-Type: application/atom+xml; charset=UTF-8
Host: smpserver:8080
```

```
Authorization: Basic <XXXX>
```

Response

Code	Description
200	No application connection entity
404	Not explicitly registered the client

1.6.1.3.5.10 Delete Application Connection

Delete an application connection.

Request

URL: `https://<mobile services host>/odata/applications/<service version>/<appid>/Connections('<appid>')`

HTTP Method: `HTTP DELETE`

Request Parameters

Parameter	Type	Description
<code><appid></code>	Mandatory	ID that uniquely identifies an application
<code><appid></code>	Mandatory	The connection ID of the application instance interacting with the service
<code><service version></code>	Mandatory	v1 onwards

Request Header Example

```
DELETE /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-
bcaa-439d-9741-660a3cb56771') HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic <XXXX>
```

Response

Code	Description
200 OK	Returns service document
404	Explicitly not registered client

1.6.1.3.5.11 Revoke User Token

Revoke all tokens registered to a specific user for an application, from the server.

Usage

You can revoke all OAuth tokens registered to a specific user from the Mobile Services cockpit. This API enables you revoke user tokens from the back end server instead.

Request

URL: [[<application-service-host>/mobileservices/service-key/odata/applications/user/<user-name>/updateRevokeTime](#) | [<mobile-services-host>/mobileservices/service-key/odata/applications/user/<user-name@domain>/updateRevokeTime](#)]

HTTP Method: *PATCH*

Request Parameters:

Parameter	Type	Description
<application-service-host>	Mandatory	The server that hosts the application.
<user-name>	Mandatory	The specific user name or identifier for which you want to revoke tokens.

HTTP Header: X-API-Key = {service key}

Service key format example:

7d5e81512b5c6f371113a73d395913e7ba8732440ac0a1f1f81833ff1dea53a5

To create a service key, see [Service Keys](#).

Note

When you create a new service key under the Mobile Settings Exchange feature, please choose the `revoke_user_token` role. If a user can get the service key, he or she can call the back-end API to revoke a user's token.

Response

Code	Description
200	Revoke specific user token success
404	Not found specific user registration
500	Server internal error

1.6.1.3.6 Error Codes and Message Formats

The server returns different formats for error codes and messages according to different "Accept" values in request headers.

Accept Header and Data Format

Type and Format	Accept Header Values	Sample Response Body
XML	application/xml, application/xhtml+xml, application/atom+xml	<pre><html><head><title>"message string"</title></head><body><h1>"error code" - "error string".</h1><p>message string <u>error string</u></p><p>description <u>error message</u></p><h3>"text string"</h3></body></html></pre>
JSON	application/json, text/json	<pre>{ "error": { "code": "403", "message": { "lang": "en-US", "value": "some specific error text string" } } }</pre>
TEXT	text/html, text/plain	"some specific error text string"

📘 Note

If the Accept header does not include any of these data types, the response body is null.

1.6.1.3.7 External Access to Application Versions

Provides third-party access to meta-data and binary information about published artifacts stored in the App Lab service. These APIs require authentication and authorization using a service-key.

📘 Note

The following documentation uses {<ArtifactsUrl>} to represent the base-URL needed to call the APIs (or endpoints). The base-URL is provided with a service-key. The response examples in this document use <{ArtifactsUrl}> for short-hand notation, but on a deployed system the "real" base-URL for the service-key is used.

Authentication and Authorization

In the Mobile Services cockpit, under ► [Settings](#) ► [App Catalog](#) ► [Service Keys](#) , administrators and developers define service keys for accessing published applications through the Mobile Services Catalog API.

{<ArtifactsUrl>} is found under "URL".

Authentication

A service-key is required to access the artifacts API. To properly authenticate with the APIs, the provided `service-key` needs to be specified in the header `x-api-key` upon each restful request. Along with this `service-key`, a base-URL will be provided; this is the value referenced by the {<ArtifactsUrl>} found throughout this document.

Authorization

The service-key may have one or more roles assigned to it. The following roles must be assigned to the service-key to access artifact endpoints:

- **list_published_artifacts**: required to be able to read meta-data.
- **download_published_artifacts**: required to be able to download content.

Artifact Meta-data Overview

The endpoint uses Open Data Protocol Version 2 (OData) to access the data. Extensive knowledge of OData is not required, and simple examples of restful queries will be provided (see the section "Simple OData restful queries"). See references for more information on OData.

Types of Artifact Meta-data Available (or OData Entity-Set)

Access to the following entities is provided via OData:

FlatApps: provides a flat (non-hierarchical) representation of all application content. The data will contain a lot of duplicated information, but offers a more simplified perspective of application data. This particular representation is easier to query/filter from a client's perspective, and doesn't require drilling down into a hierarchical representation.

Noteworthy Properties/Fields

Most of the properties/fields are self-descriptive, but a few warrant a little bit more discussion, specifically the `Url`, `Uri` and `Type` properties. These properties are reported as part of the `FlatApps`, `AppBlobs` and `AppVersionBlobs` entities.

- **Url**: specifies the full URL which can be used directly to download the binary content.
Uri: specifies the URI which should be appended to {<ArtifactsUrl>}; the resultant URL can be used to download the binary content. {<ArtifactsUrl>} will be provided as part of the service-key.
Type: designates the content type. Currently two types of content are specified:
 - **appBinary**: the application archive (apk or ipa).
 - **appLargeIcon**: the associated icon for the application archive.

Artifacts Meta-data Endpoint

The base-URL for the endpoint is: {<ArtifactsUrl>}/appcat/v1/artifacts/odata

Path and query parameters need to be added to this URL to access specific information.

A service-key is required to access this endpoint. The service-key will need to have the role `list_published_artifacts` assigned to it in order to successfully access this endpoint.

Below we provide some very basic restful queries which can be used to retrieve OData schema/meta-data, entity-sets available, as well data associated with an entity-set. These basic queries can be used to verify you can access the artifacts meta-data successfully, using the provided service-key. See the section Simple OData restful queries for more practical queries.

Example: retrieving a collection

Description: Retrieve a flat representation of ALL application content, with the response formatted as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?\$format=json

Response:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "type": "appcat.FlatApp"
        },
        "Path": "sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2FReader_App_9.0.2-1.apk",
        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Name": "sample.test.app",
        "DisplayName": "sampleTestApp",
        "Description": "sample test app",
        "VersionNotes": "test",
        "ApplicationID": "com.sample.readerapp",
        "Platform": "Android",
        "Version": "9.0.2",
        "BuildVersion": "33187",
        "FormFactors": null,
        "Type": "appBinary",
        "Size": 1839413,
        "CreatedDate": "/Date(1544807746000)/",
        "LastModifiedDate": "/Date(1546455227100)/"
      },
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "type": "appcat.FlatApp"
        },
        "Path": "sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",
        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Name": "sample.test.app",
        "DisplayName": "sampleTestApp",
        "Description": "sample test app",
        "VersionNotes": "test",
        "ApplicationID": "com.sample.readerapp",
        "Platform": "Android",
        "Version": "9.0.2",
        "BuildVersion": "33187",
        "FormFactors": null,
        "Type": "appBinary",
        "Size": 1839413,
        "CreatedDate": "/Date(1544807746000)/",
        "LastModifiedDate": "/Date(1546455227100)/"
      }
    ]
  }
}
```

```

        "type": "appcat.FlatApp"
      },
      "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",
      "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
      "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
      "Name": "sample.test.app",
      "DisplayName": "sampleTestApp",
      "Description": "sample test app",
      "VersionNotes": "test",
      "ApplicationID": "com.sample.readerapp",
      "Platform": "Android",
      "Version": "9.0.2",
      "BuildVersion": "33187",
      "FormFactors": null,
      "Type": "appLargeIcon",
      "Size": 3368,
      "CreatedDate": "/Date(1544807745099)/",
      "LastModifiedDate": "/Date(1546455227100)/"
    }
  ]
}

```

Artifacts Binary (blob) Endpoint

The base-URL for the binary endpoint is: `{<ArtifactsUrl>}/appcat/v1/artifacts/blob`

A `service-key` is required to access this endpoint. The `service-key` will need to have the role `download_published_artifacts` assigned to it in order to successfully access this endpoint.

Path parameters will need to be added to this URL to retrieve specific content associated with an artifact. As discussed previously (see the "Artifact meta-data overview" discussion), the artifact meta-data property for `Uri` (in combination with `{<ArtifactsUrl>}`) will provide the URL necessary to access the binary content successfully.

Noteworthy information about this endpoint:

- GET and HEAD are the only supported methods on this endpoint.
 - HEAD request should be used to retrieve information about the content without downloading content.
 - GET request should be used to download content.
- Range requests: if advertised, range requests are supported. Range requests can be used to resume download of partially downloaded content. The use of range requests is recommended, specifically to download the content in "chunks"; if the download is interrupted due to latency or poor network conditions, the download may be resumed by use of range headers. To determine if range requests are supported:
 - Perform a HEAD request on the specified content before initiating a download.
 - Determine if `Accept-Ranges` header is present:
 - If the header is present and set to "bytes", then range requests are supported.
 - If the header is present and set to "none", then range requests are NOT supported.
 - If the header is not present, then range requests are NOT supported.
 - For further discussion on range request, see the following link: [Range Requests](#) ➡

Example: Retrieving Binary Content

Description: Retrieve a binary content. {<blobUriPath>} is specific to the content being downloaded.


Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/blob/{<blobUriPath>}

Response:

Not shown (binary)

Simple OData restful queries:

This section provides examples of simple OData queries. This is a starting point for user wanting to access the artifacts meta-data without the use of an OData client.

For detailed information of the various query parameters used in this section, see [OData Query Parameters](#) ; this reference document contains many useful examples regarding the use of query parameters. This section will only cover a subset of query parameters available by OData; please refer to the documentation for information on other query parameters. As appropriate, links to additional information on the specific OData query parameter will be provided.

EXAMPLE: Specifying Response Format (\$format: [Format Query Parameter](#))

OData supports the following response formats:

- XML format (default)
- JSON format

To specify the desired format, use the query parameter \$format and set it to one of the following values:

- json (for JSON format)
- xml (for XML format)

Note

This query parameter is applicable to most OData queries. By default the OData response will default to XML format if this parameter is not specified.

Example

Description: Retrieve a flat representation of ALL application content, with the response formatted as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?\$format=json

Response:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
```

```

        "type": "appcat.FlatApp"
      },
      "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2FReader_App_9.0.2-1.apk",
      "Uri": "/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
      "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
      "Name": "sample.test.app",
      "DisplayName": "sampleTestApp",
      "Description": "sample test app",
      "VersionNotes": "test",
      "ApplicationID": "com.sample.readerapp",
      "Platform": "Android",
      "Version": "9.0.2",
      "BuildVersion": "33187",
      "FormFactors": null,
      "Type": "appBinary",
      "Size": 1839413,
      "CreatedDate": "/Date(1544807746000)/",
      "LastModifiedDate": "/Date(1546455227100)/"
    },
    {
      "__metadata": {
        "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
        "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
        "type": "appcat.FlatApp"
      },
      "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",
      "Uri": "/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/icon.png",
      "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/icon.png",
      "Name": "sample.test.app",
      "DisplayName": "sampleTestApp",
      "Description": "sample test app",
      "VersionNotes": "test",
      "ApplicationID": "com.sample.readerapp",
      "Platform": "Android",
      "Version": "9.0.2",
      "BuildVersion": "33187",
      "FormFactors": null,
      "Type": "appLargeIcon",
      "Size": 3368,
      "CreatedDate": "/Date(1544807745099)/",
      "LastModifiedDate": "/Date(1546455227100)/"
    }
  ]
}

```

EXAMPLE: Returning a subset of data (\$select: [Select Query Parameter](#) ➦)

By default, an OData query will return all properties associated with an entity. This query parameter permits the ability to retrieve a subset of data. To specify a subset of properties returned in a query, use the query parameter \$select.

Example

Description: Retrieve a collection of FlatApps, but only report the Url and Type properties, with the response formatted as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?
\$format=json&\$select=Type,Url

Response:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "type": "appcat.FlatApp"
        },
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Type": "appBinary",
      },
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "type": "appcat.FlatApp"
        },
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Type": "appLargeIcon",
      }
    ]
  }
}
```

EXAMPLE: Filtering data (\$filter: [Filter Query Parameter](#) 🦋)

By default, an OData query will not filter any data associated with a collection. This query parameter permits the ability to filter collection data. To specify a filter for a query, use the query parameter `$filter`.

📌 Note

The following example is simple. Options available for the `$filter` parameter are quite extensive; please refer to the documentation for this query parameter

Example 1

Description: Retrieve a collection of FlatApps but only include those items having value of "appBinary" for the Type property; format the response as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?
\$format=json&\$filter=Type eq 'appBinary'

Response:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252FReader_App_9.0.2-1.apk')",
          "type": "appcat.FlatApp"
        },
        "Path": "sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2FReader_App_9.0.2-1.apk",
        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Name": "sample.test.app",
        "DisplayName": "sampleTestApp",
        "Description": "sample test app",
        "VersionNotes": "test",
        "ApplicationID": "com.sample.readerapp",
        "Platform": "Android",
        "Version": "9.0.2",
        "BuildVersion": "33187",
        "FormFactors": null,
        "Type": "appBinary",
        "Size": 1839413,
        "CreatedDate": "/Date(1544807746000)/",
        "LastModifiedDate": "/Date(1546455227100)/"
      }
    ]
  }
}
```

Example 2

Description: Retrieve a collection of FlatApps but only include those items having value of "appLargeIcon" for the Type property; format the response as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?
\$format=json&\$filter=Type eq 'appLargeIcon'

Response:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%252Ficon.png')",
          "type": "appcat.FlatApp"
        },
        "Path": "sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",

```



```

        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Name": "sample.test.app",
        "DisplayName": "sampleTestApp",
        "Description": "sample test app",
        "VersionNotes": "test",
        "ApplicationID": "com.sample.readerapp",
        "Platform": "Android",
        "Version": "9.0.2",
        "BuildVersion": "33187",
        "FormFactors": null,
        "Type": "appLargeIcon",
        "Size": 3368,
        "CreatedDate": "/Date(1544807745099)/",
        "LastModifiedDate": "/Date(1546455227100)/"
    }
}
]
}
}

```

EXAMPLE: Ordering data (\$orderby: [Orderby Query Parameter](#) ➡)

This query parameter permits the ability order collection data. To specify ordering of data, use the query parameter \$orderby. If ascending (asc) or descending (desc) ordering is not specified with this query parameter, the ordering is performed in ascending order.

Example 1

Description: Retrieve a collection of FlatApps, ordering the collection in ascending (asc) order based upon the CreatedDate property; format the response as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?
\$format=json&\$orderby=CreatedDate asc

Response (excerpt):

```

{
  "d": {
    "results": [
      {
        "__metadata": {
          "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2Ficon.png')",
          "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2Ficon.png')",
          "type": "appcat.FlatApp"
        },
        "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",
        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
        "Name": "sample.test.app",
        ...
        "Type": "appLargeIcon",
        "Size": 3368,
        "CreatedDate": "/Date(1544807745099)/",
        "LastModifiedDate": "/Date(1546455227100)/"
      },
    ]
  }
}

```

```

        "__metadata": {
            "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2FReader_App_9.0.2-1.apk')",
            "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2FReader_App_9.0.2-1.apk')",
            "type": "appcat.FlatApp"
        },
        "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2FReader_App_9.0.
2-1.apk",
        "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
        ...
        "Type": "appBinary",
        "Size": 1839413,
        "CreatedDate": "/Date(1544807746000)/",
        "LastModifiedDate": "/Date(1546455227100)/"
    }
}
}
}

```

Example 2

Description: Retrieve a collection of FlatApps, ordering the collection in descending (desc) order based upon the CreatedDate property; format the response as json.

Request: GET {<ArtifactsUrl>}/appcat/v1/artifacts/odata/FlatApps?
\$format=json&\$orderby=CreatedDate desc

Response (excerpt):

```

{
    "d": {
        "results": [
            {
                "__metadata": {
                    "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2FReader_App_9.0.2-1.apk')",
                    "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2FReader_App_9.0.2-1.apk')",
                    "type": "appcat.FlatApp"
                },
                "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2FReader_App_9.0.
2-1.apk",
                "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
                "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/Reader_App_9.0.2-1.apk",
                ...
                "Type": "appBinary",
                "Size": 1839413,
                "CreatedDate": "/Date(1544807746000)/",
                "LastModifiedDate": "/Date(1546455227100)/"
            },
            {
                "__metadata": {

```

```

        "id": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2Ficon.png')",
        "uri": "{ArtifactsUrl}/appcat/v1/artifacts/odata/
FlatApps('sample.test.app%252Fcom.sample.readerapp%252FAndroid%252F9.0.2_33187%25
2Ficon.png')",
        "type": "appcat.FlatApp"
    },
    "Path":
"sample.test.app%2Fcom.sample.readerapp%2FAndroid%2F9.0.2_33187%2Ficon.png",
    "Uri": "/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
    "Url": "{ArtifactsUrl}/appcat/v1/artifacts/blob/sample.test.app/
com.sample.readerapp/Android/9.0.2_33187/icon.png",
    "Name": "sample.test.app",
    ...
    "Type": "appLargeIcon",
    "Size": 3368,
    "CreatedDate": "/Date(1544807745099)/",
    "LastModifiedDate": "/Date(1546455227100)/"
}
    ]
}
}

```

References

OData Version 2

- Getting started with OData: [OData Getting Started](#) ➦
- OData Version2 Documentation: [OData Version2 Document](#) ➦
 - OData Query Parameters: [OData Query Parameters](#) ➦
 - \$format: [Format Query Parameter](#) ➦
 - \$select: [Select Query Parameter](#) ➦
 - \$filter: [Filter Query Parameter](#) ➦
 - \$orderby: [Orderby Query Parameter](#) ➦
 - \$top: [Top Query Parameter](#) ➦
 - \$skip: [Skip Query Parameter](#) ➦
 - \$expand: [Expand Query Parameter](#) ➦
 - \$inlinecount: [Inlinecount Query Parameter](#) ➦

Range Requests

Information on range requests: [Range Requests](#) ➦

1.6.1.3.8 Authenticate Applications Using SAML 2.0

Initiate a REST service call to create SAML 2.0 assertion for authenticating the application security configuration.

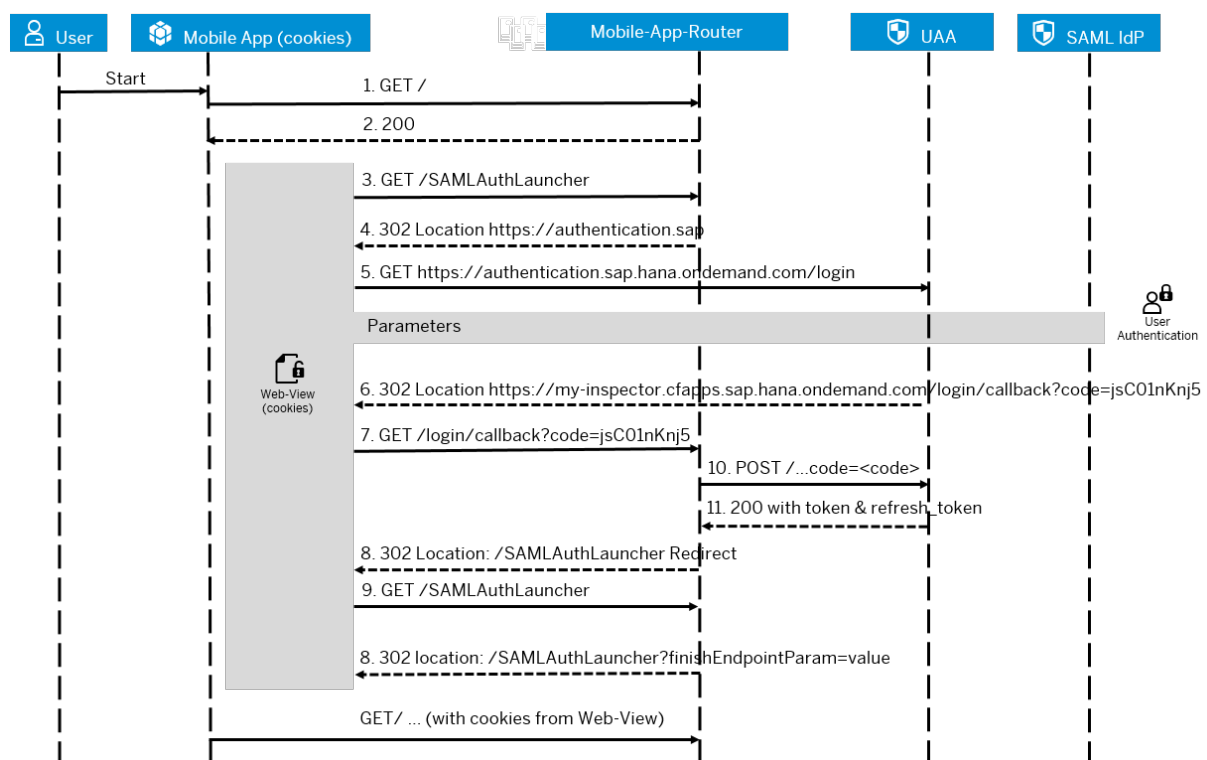
Usage

When an application initially connects to the server, a session is established. If the application is set up to be secured by SAML 2.0 authentication, the server responds with the header `com.sap.cloud.security.login:login-request` and SAML 2.0 authentication for the security configuration needs to take place in the application.

Note

This mechanism is also followed for any session that has not been authenticated, or has expired.

For Mobile Services on Cloud Foundry, SAML 2.0 provides an authentication flow that uses headers and cookies, which is compatible with the Neo SAML/FORM authentication flow.



Request

Issue an HTTP request to the server. If the server responds, the header indicates that SAML 2.0 authentication is required.

URL: http[s]://<mobile services host>/SAMLAuthLauncher

HTTP Method: *GET*

Request Parameters None

Request Body Example

1. When an application is initially launched, it sends a request that establishes a connection with the server. If the application is secured by SAML 2.0 authentication, the server sends a response containing these elements:

- Response Header:
 - Name: com.sap.cloud.security.login
 - Value: login-request
- Cookie JSESSIONID (no session cookie is returned for the first request)
- Status Code: HTTP-OK – 200

Ensure that the response header contains the name and value `com.sap.cloud.security.login: login-request`, which indicates that SAML 2.0 authentication is required. If the response header is not returned, authentication does not take place.

HTTP request: there is no requirement for the initial request that is sent to the server. The request can be directed to any server resources.

HTTP response header for the initial request:

```
com.sap.cloud.security.login: login-request
Date: Mon, 25 Mar 2019 03:10:12 GMT X-Smp-Log-Correlation-Id:
56f007db-446c-4d82-7b3b-57eafd108715
X-Vcap-Request-Id: 56f007db-446c-4d82-7b3b-57eafd108715
Content-Length: 1062
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload;
```

2. When the response is received, the application starts the authentication process, using the web view.

```
/*Now that you have received com.sap.cloud.security.login: login-request
response header and SAML2 JavaScript redirect in the response body
.
*/
Issue a GET method on the request URL:
GET https://mobiletest-smoketest-testssaml.cfapps.eu10.hana.ondemand.com/
SAMLAuthLauncher
```

Request headers: the request to "https://mobiletest-smoketest-testssaml.cfapps.eu10.hana.ondemand.com/SAMLAuthLauncher" requires no special request header requirements or restrictions.

Response headers: since the request is processed by web view, the client application does not need to process this response by itself.

Response body: includes a JavaScript code to which to redirect, in this example to:

```
https://mobiletest.authentication.eu10.hana.ondemand.com/oauth/
authorize?response_type=code&client_id=sb-testssaml-smoketest-
kxx0ni6n!t632&redirect_uri=https%3A%2F%2Fmobiletest-smoketest-
testssaml.cfapps.eu10.hana.ondemand.com%2Flogin%2Fcallback
```

Response Code: 200

Response: N/A

3. To complete SAML 2.0 authentication, the following operation takes place automatically:

1. The web view is redirected to the UAA login URL. The UAA may also redirect web view to a SAML 2.0 identity provider sign-on login URL, depending on the configuration of trusted identity providers for the customer subaccount.
2. After successful login, the web view is redirected back to the mobile application to check the response from UAA at:

```
<host:port>/login/callback
```

3. The mobile application checks the response and creates an authenticated session for the application. The web view is redirected to:

```
<host:port>/SAMLAuthLauncher?finishEndpointParam=someUnusedValue
```

```
/*After successful authentication on the UAA, you are redirected to the /
login/callback endpoint of the mobile application with an authorization code.
*/A GET method is issued on the request URL: GET https://mobiletest-smoketest-
testsaml.cfapps.eu10.hana.ondemand.com/login/callback?code=dV3hXO2AsO
```

Request header:

```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Encoding: gzip, deflate, br
Accept-Language: en
Connection: keep-alive
Cookie: locationAfterLogin=%2FSAMLAuth...sedValue; fragmentAfterLogin=
Host: mobiletest-smoketest-testsaml.cfapps.eu10.hana.ondemand.com
Referer: https://mobiletest.authenticat...n.eu10.hana.ondemand.com/login
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/66.0
```

Response headers:

```
Cache-Control: no-cache, no-store, must-revalidate
Content-Length: 0
Date: Mon, 25 Mar 2019 03:34:29 GMT
Location: /SAMLAuthLauncher?finishEndpointParam=someUnusedValue
Set-Cookie: locationAfterLogin=; Max-Age=0; Path=/
Set-Cookie: fragmentAfterLogin=; Max-Age=0; Path=/ Set-Cookie:
JSESSIONID=s%3AIRlg1MdJDRkbutF...ItA; Path=/; HttpOnly; Secure
Set-Cookie: __VCAP_ID__=4cd4a91c-2690-424c...54d; Path=/; HttpOnly; Secure
Strict-Transport-Security: max-age=31536000; includeSubDomains; preload;
X-Frame-Options: SAMEORIGIN
X-Smp-Log-Correlation-Id: 3dae91a6-6b72-4091-6dc7-b962fd8971ce
X-Vcap-Request-Id: 3dae91a6-6b72-4091-6dc7-b962fd8971ce
```

Response code: 302

4. After the web view is redirected, close the view, then invoke the original REST service call by using the authenticated session (cookie) from the web view.

Re-send the registration request.

Request:

```
POST https://mobiletest-smoketest-testsaml.cfapps.eu10.hana.ondemand.com/odata/
applications/latest/testsaml/Connections
```

Request Payload:

```
<?xml version='1.0' encoding='utf-8'?>
  <entry xmlns="http://www.w3.org/2005/Atom">
```

```

    xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
    xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
    <title type="text"/>
    <updated>2012-06-15T02:23:29Z</updated>
    <author>
    <name/>
    </author>
    <category term="applications.Connection" scheme="http://
schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
    <content type="application/xml">
    <m:properties>
    <d:DeviceType>iPad</d:DeviceType>
    <d:DeviceModel m:null="true" />
    </m:properties>
    </content>
    </entry>
Status 201 Created

```

Request headers:

```

Content-Type: application/atom+xml
Cookie:
JSESSIONID=s%3AIRlg1MdJDRkbutFiE2Sbrxc4YTS1b_Pb.y6ptKBimMH%2FokntVEAu%2FF1DfGM2ui
Ezczq0yTejRItA; __VCAP_ID__=4cd4a91c-2690-424c-6fc9-754d

```

Response headers:

```

Set-Cookie: X-SMP-APPCID=b46df728-c5d8-4c03-a175-71f7a496280e;
Content-Type: application/atom+xml; charset=utf-8

```

Response:

```

<?xml version="1.0" encoding="utf-8"?><entry xmlns="http://www.w3.org/2005/Atom"
xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices"
xml:base="https://mobileciathanamobile-x054703e3.neo.ondemand.com/odata/
applications/latest/com.sap.maf.test_SAML2/">
<id>https://mobileciathanamobile-x054703e3.neo.ondemand.com/odata/
applications/latest/com.sap.maf.test_SAML2/Connections('b46df728-c5d8-4c03-
a175-71f7a496280e')</id>
<title type="text"></title><updated>2015-01-19T08:44:20Z</
updated><author><name></name></author>
<link rel="edit" title="Connection" href="Connections('b46df728-c5d8-4c03-
a175-71f7a496280e')"></link>
<category term="applications.Connection" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"></category>
<content type="application/xml"><m:properties><d:ETag>2015-01-19 08:44:20.0</
d:ETag>
<d:ApplicationConnectionId>b46df728-c5d8-4c03-a175-71f7a496280e</
d:ApplicationConnectionId>
<d:AndroidGcmPushEnabled m:type="Edm.Boolean">false</d:AndroidGcmPushEnabled>
<d:AndroidGcmRegistrationId m:null="true"></d:AndroidGcmRegistrationId>
<d:AndroidGcmRegistrationId><d:AndroidGcmSenderId></d:AndroidGcmSenderId>
<d:ApnsPushEnable m:type="Edm.Boolean">false</
d:ApnsPushEnable><d:ApnsDeviceToken m:null="true"></
d:ApnsDeviceToken><d:ApplicationVersion>1.0</d:ApplicationVersion>
<d:BlackberryPushEnabled m:type="Edm.Boolean">false</
d:BlackberryPushEnabled><d:BlackberryDevicePin m:null="true"></
d:BlackberryDevicePin>
<d:BlackberryBESListenerPort m:type="Edm.Int32">0</
d:BlackberryBESListenerPort><d:BlackberryPushAppID m:null="true"></
d:BlackberryPushAppID>

```

```

<d:BlackberryPushBaseUrl m:null="true"></
d:BlackberryPushBaseUrl><d:BlackberryPushListenerPort m:type="Edm.Int32">0</
d:BlackberryPushListenerPort>
<d:BlackberryListenerType m:type="Edm.Int32">0</
d:BlackberryListenerType><d:CollectClientUsageReports m:type="Edm.Boolean">true</
d:CollectClientUsageReports>
<d:ConnectionLogLevel>NONE</d:ConnectionLogLevel><d:CustomizationBundleId
m:null="true"></d:CustomizationBundleId>
<d:CustomCustom1></d:CustomCustom1><d:CustomCustom2></
d:CustomCustom2><d:CustomCustom3></d:CustomCustom3><d:CustomCustom4></
d:CustomCustom4>
<d:DeviceModel m:null="true"></d:DeviceModel><d:DeviceType>iPad</
d:DeviceType><d:DeviceSubType m:null="true"></d:DeviceSubType>
<d:DevicePhoneNumber m:null="true"></d:DevicePhoneNumber><d:DeviceIMSI
m:null="true"></d:DeviceIMSI><d:E2ETraceLevel>Low</d:E2ETraceLevel>
<d:EnableAppSpecificClientUsageKeys m:type="Edm.Boolean">false</
d:EnableAppSpecificClientUsageKeys>
<d:FeatureVectorPolicyAllEnabled m:type="Edm.Boolean">true</
d:FeatureVectorPolicyAllEnabled>
<d:LogEntryExpiry m:type="Edm.Int32">7</
d:LogEntryExpiry><d:MaxConnectionWaitTimeForClientUsage
m:type="Edm.Boolean">false</d:MaxConnectionWaitTimeForClientUsage>
<d:MpnsChannelURI m:null="true"></d:MpnsChannelURI><d:MpnsPushEnable
m:type="Edm.Boolean">false</d:MpnsPushEnable>
<d>PasswordPolicyEnabled m:type="Edm.Boolean">false</
d>PasswordPolicyEnabled><d>PasswordPolicyDefaultPasswordAllowed
m:type="Edm.Boolean">false</d>PasswordPolicyDefaultPasswordAllowed>
<d>PasswordPolicyMinLength m:type="Edm.Int32">8</
d>PasswordPolicyMinLength><d>PasswordPolicyDigitRequired
m:type="Edm.Boolean">false</d>PasswordPolicyDigitRequired>
<d>PasswordPolicyUpperRequired m:type="Edm.Boolean">false</
d>PasswordPolicyUpperRequired><d>PasswordPolicyLowerRequired
m:type="Edm.Boolean">false</d>PasswordPolicyLowerRequired>
<d>PasswordPolicySpecialRequired m:type="Edm.Boolean">false</
d>PasswordPolicySpecialRequired><d>PasswordPolicyExpiresInNDays
m:type="Edm.Int32">0</d>PasswordPolicyExpiresInNDays>
<d>PasswordPolicyMinUniqueChars m:type="Edm.Int32">0</
d>PasswordPolicyMinUniqueChars><d>PasswordPolicyLockTimeout
m:type="Edm.Int32">0</d>PasswordPolicyLockTimeout>
<d>PasswordPolicyRetryLimit m:type="Edm.Int32">20</
d>PasswordPolicyRetryLimit><d:ProxyApplicationEndpoint>https://
vmw3815.wdf.sap.corp:44309/sap/opu/odata/GBHCM/LEAVEREQUEST/</
d:ProxyApplicationEndpoint>
<d:ProxyPushEndpoint m:null="true"></
d:ProxyPushEndpoint><d:PublishedToMobilePlace m:type="Edm.Boolean">false</
d:PublishedToMobilePlace>
<d:UploadLogs m:type="Edm.Boolean">true</d:UploadLogs><d:WnsChannelURI
m:null="true"></d:WnsChannelURI>
<d:WnsPushEnable m:type="Edm.Boolean">false</
d:WnsPushEnable><d:FeatureVectorPolicy
m:type="Bag(applications.FeatureVectorPolicy)"></d:FeatureVectorPolicy>
</m:properties></content></entry>

```

Note

At any point when the SAML2 session is invalid, or the binding cookies on the client side expire, you must encounter SAML2 form response.

1.6.1.3.9 Retrieve Customization Resource Bundles

Download application resource bundles.

Note

Application developers can customize and retrieve resource bundles.. If the values of `<resourceBundleName>` and `<resourceBundleVersion>` are specified in the URL, the resource bundle is returned in the response body as a stream; otherwise, the resource bundle that is bound to the application is returned. The resource-bundle extension is in the response header X-BUNDLE-EXTENSION.

If the resource bundle is not found in mobile platform, error code 404 is returned. You cannot issue other HTTP methods (PUT/POST/DELETE) at the above URL.

Request

URL: `https://<mobile services host>/bundles/<appid>/[<resourceBundleName>:<Version>]`

HTTP Method: `GET`

Request Parameters

Parameter	Type	Description
appid	Mandatory	ID that uniquely identifies an application
resourceBundleName	Optional	Returns the resource bundle
Version	Optional	Returns version of the resource bundle

Request Body Example

```
GET /bundles/com.sap.myapp/MyApp:1.0 HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Host: smpserver:8080
Authorization: Basic <XXXX>
```

Response

Code	Description
200 OK	Returns resource bundle content
404 Not Found	Resource bundle is not found

1.6.1.3.10 Accessing Services Through Proxy URLs

To access a back end or Internet-based service, use a proxy URL that supports read, create, update, delete, merge and patch.

Note

Verify that all the URLs to be proxied are whitelisted.

Usage

You can specify the customized application properties for client requests. Provide the application connection ID (X-SMP-APPCID) by using an explicit request header or a cookie.

HTTP Operations

HTTP Method	Request URL	Description
<i>GET</i>	<code>https://<mobile application host>/{connectionName}/ [<Collection>]</code>	Retrieve data from the back end through the mobile platform.
<i>POST</i>	<code>https://<mobile application host>/{connectionName}/ [<Collection>]</code>	Requests the server to accept the data in the request message body.
<i>PUT</i>	<code>https://<mobile application host>/{connectionName}/ [<Collection>]/ ('<EntryID>')</code>	Update an entry in the back end.
<i>DELETE</i>	<code>https://<mobile application host>/{connectionName}/ [<Collection>]/ ('<EntryID>')</code>	Delete an entry from the back end.

HTTP Method	Request URL	Description
<i>MERGE</i>	<code>https://<mobile application host>/{connectionName}/[Collection]/(' <EntryID> ')</code>	Incrementally updates without replacing all the content of an entry.
<i>PATCH</i>	<code>https://<mobile application host>/{connectionName}/[Collection]/(' <EntryID> ')</code>	Performs partial updates without replacing all the content of an entry. A PATCH request updates only the properties indicated in the request body.

Request Header Example

```
X-SMP-APPCID : <Application connection Id received in the response of the
onboarding xml>
Content-Type : application/atom+xml
X-Requested-With : XMLHttpRequest
Authorisation : <Base 64 encoded value of Authorization>
```

Response

HTTP Method	Code	Description
GET	200	Returns data from back end
POST	201	Returns when server accepts the data
PUT	204	Returns on successful update of entry in the back end
DELETE	204	Returns on successful deletion of entry in the back end
<div> <div>📌 Note</div> <div>No information is returned from a DELETE request.</div> </div>		
MERGE	204	Returns on successful merge of entry in the back end
PATCH	204	Returns on successful merge of entry in the back end

Related Information

[Custom Push \[page 147\]](#)

1.6.1.3.11 Feature Restriction Policies

REST API methods for managing feature restriction policies for an application. You can get, update, or remove features enabled through the Java API, `isEnabled()`. Any enabled feature can be disabled by the administrator through the cockpit, providing additional control.

[Get Feature Restriction Policy \[page 436\]](#)

Get the feature restriction (or vector) policy for an application.

[Update Feature Restriction Policy \[page 437\]](#)

Update the feature restriction (or vector) policy for an application.

[Remove Feature Restriction Policy \[page 439\]](#)

Remove a feature (or vector) restriction policy from an application.

Related Information

[Supported Onboarding Services \[page 397\]](#)

[Create Application Connection with Capability Handling \[page 403\]](#)

[Create Application Connection \[page 402\]](#)

1.6.1.3.11.1 Get Feature Restriction Policy

Get the feature restriction (or vector) policy for an application.

Request

URL: `https://<mobile services host>/mobileservices/Storage/v1/runtime/application/{<appid>}/global/mobileservices/settingsExchange/featureVectorPolicies`

HTTP Method: *GET*

Request Parameters

Parameter	Type	Description
appid	Mandatory	ID that uniquely identifies an application.

Request Header Example

```
Content-Type: application/json
Authorization: Business User Token
```

Response

```
{
  "restrictedPolicies": [
    {
      "pluginName": "Camera",
      "displayName": "Camera",
      "name": "Camera",
      "description": "Cordova Camera Plugin",
      "id": "cordova-plugin-camera.Camera",
      "whitelist": "*",
      "jsModule": "Camera,navigator.camera,CameraPopoverOptions,CameraPopoverHandle",
      "version": "3.0"
    }
  ],
  "allowedPolicies": [
    {
      "pluginName": "Contacts",
      "displayName": "Contacts",
      "name": "Contacts",
      "description": "Cordova Contacts Plugin",
      "id": "cordova-plugin-contacts.Contact",
      "whitelist": "*",
      "jsModule":
        "navigator.contacts,Contact,ContactAddress,ContactError,ContactField,ContactFindOptions,ContactName,ContactOrganization",
      "version": "3.0"
    },
    {
      "pluginName": "File",
      "displayName": "File",
      "name": "File",
      "description": "Cordova File Plugin",
      "id": "cordova-plugin-file.File",
      "whitelist": "*",
      "jsModule":
        "window.DirectoryEntry,window.DirectoryReader,window.Entry,window.File,window.FileEntry,window.FileError,window.FileReader,window.FileSystem,window.FileUploadOptions,window.FileUploadResult,window.FileWriter,window.Flags,window.LocalFileSystem,window.Metadata,window.ProgressEvent,window.requestFileSystem",
      "version": "3.0"
    }
  ],
  "featureVectorPolicyAllEnabled": "false"
}
```

1.6.1.3.11.2 Update Feature Restriction Policy

Update the feature restriction (or vector) policy for an application.

Request

URL: `https://<mobile services host>/mobileservices/Storage/v1/runtime/application/<appid>/global/mobileservices/settingsExchange/featureVectorPolicies`

HTTP Method: `PUT/PATCH`

Request Parameters

Parameter	Type	Description
appid	Mandatory	ID that uniquely identifies an application.

Request Header Example

```
Content-Type: application/json
Authorization: Business User Token
```

Note

This API requires the Administrator Role. By default, business users are not assigned this role.

Response

Response example:

```
{
  "restrictedPolicies": [
    {
      "pluginName": "Camera",
      "displayName": "Camera",
      "name": "Camera",
      "description": "Cordova Camera Plugin",
      "id": "cordova-plugin-camera.Camera",
      "whitelist": "*",
      "jsModule": "Camera,navigator.camera,CameraPopoverOptions,CameraPopoverHandle",
      "version": "3.0"
    }
  ],
  "allowedPolicies": [
    {
      "pluginName": "Contacts",
      "displayName": "Contacts",
      "name": "Contacts",
      "description": "Cordova Contacts Plugin",
      "id": "cordova-plugin-contacts.Contact",
      "whitelist": "*",
      "jsModule": "navigator.contacts,Contact,ContactAddress,ContactError,ContactField,ContactFindOptions,ContactName,ContactOrganization",
      "version": "3.0"
    },
    {
      "pluginName": "File",
      "displayName": "File",
      "name": "File",
      "description": "Cordova File Plugin",
      "id": "cordova-plugin-file.File",
      "whitelist": "*",
      "jsModule": "window.DirectoryEntry,window.DirectoryReader,window.Entry,window.File,window.FileEntry,window.FileError,window.FileReader,window.FileSystem,window.FileUploadOptions,window.FileUploadResult,window.FileWriter,window.Flags,window.LocalFileSystem,window.Metadata,window.ProgressEvent,window.requestFileSystem",
      "version": "3.0"
    }
  ]
}
```

```

    },
    "featureVectorPolicyAllEnabled": "false"
  }
}

```

1.6.1.3.11.3 Remove Feature Restriction Policy

Remove a feature (or vector) restriction policy from an application.

Request

URL: `https://<mobile_services_host>/mobileservices/Storage/v1/runtime/application/<appid>/global/mobileservices/settingsExchange/featureVectorPolicies`

HTTP Method: `DELETE`

Request Parameters

Parameter	Type	Description
appid	Mandatory	ID that uniquely identifies an application.

Request Header Example

```

Content-Type: application/json
Authorization: Business User Token

```

Note

This API requires the Administrator Role. By default, business users are not assigned this role.

Response

```

{
  "restrictedPolicies": [
    {
      "pluginName": "Camera",
      "displayName": "Camera",
      "name": "Camera",
      "description": "Cordova Camera Plugin",
      "id": "cordova-plugin-camera.Camera",
      "whitelist": "*",
      "jsModule": "Camera,navigator.camera,CameraPopoverOptions,CameraPopoverHandle",
      "version": "3.0"
    }
  ],
  "allowedPolicies": [
    {

```

```

"pluginName": "Contacts",
"displayName": "Contacts",
"name": "Contacts",
"description": "Cordova Contacts Plugin",
"id": "cordova-plugin-contacts.Contact",
"whitelist": "*",
"jsModule":
"navigator.contacts,Contact,ContactAddress,ContactError,ContactField,ContactFindOptions,ContactName,ContactOrganization",
"version": "3.0"
},
{
"pluginName": "File",
"displayName": "File",
"name": "File",
"description": "Cordova File Plugin",
"id": "cordova-plugin-file.File",
"whitelist": "*",
"jsModule":
"window.DirectoryEntry,window.DirectoryReader,window.Entry,window.File,window.FileEntry,window.FileError,window.FileReader,window.FileSystem,window.FileUploadOptions,window.FileUploadResult,window.FileWriter,window.Flags,window.LocalFileSystem,window.Metadata,window.ProgressEvent,window.requestFileSystem",
"version": "3.0"
}
],
"featureVectorPolicyAllEnabled": "false"
}

```

1.6.1.3.12 Upload Logs and Traces

Upload client logs and Business Transaction XML (BTX) files for analysis.

Usage

SAP Mobile Services provides a generic REST service for uploading client log files, BTX, and other trace files to the database.

In the cockpit, the administrator enables log upload settings for application connections to view the logs and traces from the cockpit.

[Upload Client Logs \[page 440\]](#)

Upload Client Logs API allows the application to upload client logs to the database.

1.6.1.3.12.1 Upload Client Logs

Upload Client Logs API allows the application to upload client logs to the database.

Upload Client Logs service supports only one Runtime API using the POST operation.

[Runtime APIs \[page 441\]](#)

The Upload Client Logs API allows the application to upload the client logs to the server for further analysis.

1.6.1.3.12.1.1 Runtime APIs

The Upload Client Logs API allows the application to upload the client logs to the server for further analysis.

Runtime API

HTTP Method	Action
POST	POST Client Log Upload Service [page 485]

1.6.1.3.12.1.1.1 POST Upload Client Logs

Upload Client Logs API allows the application to upload the client logs to the server for further analysis. This method allows you to upload client logs for an application. If the client log upload option is enabled for the specified application, then the client logs get uploaded.

HTTP Method: [POST](#).

URL: `https://<mobile services host>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application identifier	String
deviceId	Optional	Device identifier. Its value will be saved with uploaded log if specified.	String

Parameter	Type	Description	Parameter Type
clientlogs	Required	Client logs	Client logs <any>Serialized Client logs. It can be zipped data, multipart data or simple binary data. Log format: ~~~#Date time#Severity#CorrelationId#Source#Location#Message# ~~~ The format of Date time is "YYYY-MM-DDThh:mm:ss.STZD", such as "2014-07-01T17:16:08.637+02:00". The valid values of Severity are DEBUG, INFO, WARN, ERROR and FATAL. The CorrelationId value is optional, and can be left as empty if client cannot provide it for each log message. The Message value is log message generated by client, and # character in it must be escaped to avoid corrupt uploading.

Response Status and Error Codes

Code	Description
204	Client logs uploaded.
400	Invalid log format.
403	Client log upload is not enabled for the application.
404	No application found.
500	Unexpected error.

1.6.1.3.13 Cross-Origin Resource Sharing Requests

The cross-origin resource sharing (CORS) standard includes HTTP headers that allow you to control resource access from Web browsers. CORS also defines a preflight HTTP OPTIONS method that requests a list of the supported methods for a resource; if the requested method is supported, the actual HTTP request is sent.

CORS Headers

To enable CORS requests, configure the header parameters. See *Configuring Cross-Origin Resource Sharing*.

Header	Description
URI Pattern	A string of characters that matches the destination.
Origin	<p>A comma-delimited list of URIs that can access the resource, for example, <code>http://example.com</code> or <code>http://*.example.com</code>.</p> <p>The default value "*" means any URI can access the resource.</p> <p>An empty value prevents cross-origin resource sharing; only URIs in the same origin can access the resource.</p>
Methods	A comma-delimited list of HTTP methods (such as GET and DELETE) that are allowed when accessing the resource.
Max Age	The number of seconds for which the results of a request can be cached. The default is 3600 seconds (60 minutes).
Allow Credentials	Always set to On. The server includes cookies when it submits requests.
Headers	<p>A comma-delimited list of HTTP request headers that you can specify in requests.</p> <div><p>Note</p><p>An empty value means any requested headers are accepted.</p></div> <p>The server includes the following headers in its responses: <code>accept</code>, <code>authorization</code>, <code>maxdataserviceversion</code>, and <code>x-smp-appid</code>.</p>
Expose Headers	A comma-delimited list of response headers that browsers can access.

Web Browser Restrictions

The browser restrictions are:

- Internet Explorer versions 9 and earlier do not support CORS-enabled browser-based applications.
- Safari supports Document Object Model (DOM) parsing with some restrictions.

Programmable HTTP Requests

Developers can send HTTP requests in AJAX, in which case, JavaScript sends the preflight HTTP OPTIONS request. The following is a jQuery AJAX example:

```
/*
Create Application Registration/Connection
*/
$.ajax({
  url: "https://hcpmshanamobile.neo.ondemand.com/odata/applications/latest/basic/Connections",
  type: "POST",
  contentType: "application/json",
  dataType: "json",
  data: JSON.stringify({
    DeviceType: "iPhone"
  }),
  success: function(data) {
    // put your code here
  },
  error: function(error) {
    // put error handling here
  },
  xhrFields: {
    withCredentials: true
  },
  crossDomain: true
});
```

Related Information

[Create Application Connection \[page 402\]](#)

1.6.1.3.14 Logout Service

The user logout service provides functionality to terminate an active user session from the client.

The service always returns HTTP status 204 (no content), even if there was no active session for the calling user at the point the call was made, and does not require authentication (but it does require a session cookie header identifying the session to be terminated). Implement the functionality using the logout service API.

Two options are available - POST and DELETE. The POST method does the same thing as DELETE, but is the option to use if the configuration includes a firewall.

[POST User Logout Request \[page 445\]](#)

Terminate an active user session from the client using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

[DELETE User Logout Request \[page 445\]](#)

Terminate an active user session from the client using the DELETE method (if the configuration includes a firewall, use the POST method instead).

Related Information

[Logging Out Users \[page 392\]](#)

1.6.1.3.14.1 POST User Logout Request

Terminate an active user session from the client using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

Request

URL: `http[s]://<mobile services host>/mobileservices/sessions/logout`

HTTP Method *POST*

Request Body Example

```
POST https://<host:port>/mobileservices/sessions/logout HTTP/1.1
Accept-Encoding: gzip,deflate
```

1.6.1.3.14.2 DELETE User Logout Request

Terminate an active user session from the client using the DELETE method (if the configuration includes a firewall, use the POST method instead).

Request

URL: `http[s]://<mobile services host>/mobileservices/sessions`

HTTP Method *DELETE*

Request Body Example

```
DELETE https://<host:port>/mobileservices/sessions HTTP/1.1
Accept-Encoding: gzip,deflate
Host: <host>
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

1.6.1.3.15 Logout V2 Service

The user logout service provides functionality to invalidate the sessions and revoke all the refresh tokens of a user.

The service always returns HTTP status 204 (no content), even if there was no active session for the calling user at the point the call was made, and does not require authentication (but it does require a session cookie header identifying the session to be terminated). Implement the functionality using the logout service API.

Two options are available - POST and DELETE. The POST method does the same thing as DELETE, but is the option to use if the configuration includes a firewall.

[POST User Logout Request \[page 446\]](#)

Invalidates the sessions using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

[DELETE User Logout Request \[page 447\]](#)

Invalidates the session using the DELETE method (if the configuration includes a firewall, use the POST method instead).

1.6.1.3.15.1 POST User Logout Request

Invalidates the sessions using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

Request

URL: `http[s]://<mobile services host>/mobileservices/sessions/v2/logout`

HTTP Method *POST*

Request Body Example

```
POST https://<host:port>/mobileservices/sessions/v2/logout HTTP/1.1
Accept-Encoding: gzip, deflate
```

1.6.1.3.15.2 DELETE User Logout Request

Invalidates the session using the DELETE method (if the configuration includes a firewall, use the POST method instead).

Request

URL: `http[s]://<mobile services host>/mobileservices/sessions/v2`

HTTP Method *DELETE*

Request Body Example

```
DELETE https://<host:port>/mobileservices/sessions/v2 HTTP/1.1
Accept-Encoding: gzip,deflate
Host: <host>
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

1.6.1.3.16 Storage Service API

Storage service facilitates application developers to persist mobile application specific data, such as user preferences, user contextual data, and application configuration.

Storage service does not contain business-specific data. Although you can create, update, or delete all the storage configurations using Runtime API and Admin API, there are some restrictions or conditions while using Runtime API.

Using Storage service, data can be stored at three levels:

- Application storage: persists global configuration and application default settings.
- User storage: persists user preferences
- Device storage: persists installation-specific or device-specific settings.

Application Storage

If you create an application storage configuration using Runtime API, then it is accessible to all users. If you get the user storage or device storage for the application with `overwrite=true` query parameter, then the application level configuration gets merged into the response.

While creating or updating application storage configuration using Admin API, a `__metadata` property (`com.sap.mobile.server.storage.admin.v1.StorageMetadata` type) can be included for the application configuration to define read and write right control of the runtime API. If no `ReadRoles` or `WriteRoles` are defined for an application configuration, then the application configuration can be accessed by all users, similar to a Runtime API.

Once the `ReadRoles` are defined for an application configuration, then the users who have the required roles can read the application configuration using Runtime API. Similarly, only these users can get the merged application configuration, while getting the user storage or device storage configuration with

`overwrite=true` query parameter using the Runtime API. Once the WriteRoles are defined for an application configuration, then the users who have the required roles can modify the application configuration using Runtime API.

The required roles are mandatory for reading or writing the application configuration using Runtime API, if an application storage configuration is created by Runtime API, but modified by Admin API to add ReadRoles or WriteRoles.

User Storage

Users can only access the user configurations that belong to them by using the Runtime API. When you get a user storage configuration with `overwrite=true` query parameter using the Runtime API, the response includes the merged application configuration, which only the users who have the required ReadRole can access it.

If a property is defined on both application storage and user storage, the one in user storage gets included in response while getting with `overwrite=true` query parameter. If no one is defined in the user storage, then the one on the application storage gets included in response while getting with `overwrite=true` query parameter.

Device Storage

User can only access the device configuration that belongs to them. While getting the device storage configuration for an application with `overwrite=true` query parameter using the Runtime API, the response includes the merged user storage configuration and application storage configuration, which only the users who have the required ReadRole can access it. The device storage property has the highest priority, and the application storage property has the lowest priority.

Samples for Storage Service

❖ Example

In most mobile applications, the configuration used in the Application Storage is used as the default settings, and the end users can customize the settings in the User Storage or Device Storage level in order to provide personalized settings for the user or device. In such cases in order to make sure that only the Administrator can modify the Application Storage configuration and other users can only view it (by Application Storage Runtime API or user/device Storage Runtime API with `"overwrite=true"` query parameter), the application storage configuration can be configured with empty ReadRoles and `["Administrator"]` WriteRoles.

❖ Example

The application storage configuration "HomeScreen" is created for application "mobileApp":

🔗 Sample Code

```
{
  "ApplicationId": "mobileApp",
  "HomeScreen": {
    "Menu": {
      "AutoHiding": false,
      "Style": "blue",
      "AutoHiding@odata.type": "Edm.Boolean",
      "Style@odata.type": "Edm.String"
    },
    "ShortCuts": ["Overview", "Profile"],
```



```

    "__metadata": {
      "ReadRoles": [],
      "WriteRoles": ["Administrator"] },
      "Menu@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
      "ShortCuts@odata.type": "Collection(Edm.String)",
      "__metadata@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageMetadata",
    },
    "HomeScreen@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject"
  }
}

```

Only users with the Administrator role can modify these settings using Runtime API, but all other users can get these settings.

Then, the mobile application can get the settings on the device using `https://<mobile services host>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/HomeScreen?overwrite=true` and fill the screen with values you received. The response body looks like:

Output Code

```

{
  "Menu": {
    "AutoHiding": false,
    "Style": "blue"
  },
  "ShortCuts": ["Overview", "Profile"]
}

```

Assume the user configures the Menu/AutoHiding property value to true at device level:

PATCH `https://<mobile services host>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/HomeScreen`

body:

Sample Code

```

{
  "Menu": {
    "AutoHiding": true
  }
}

```

The response body of `https://<mobile services host>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/HomeScreen?`

Output Code

```

overwrite=true becomes:
{
  "Menu": {
    "AutoHiding": true,
    "Style": "blue"
  },
  "ShortCuts": ["Overview", "Profile"]
}

```

If you want to provide application level settings only to particular users and not all users (or settings provisioned in mobile app), then the ReadRoles can be defined, so that only users who have the Read role can get these settings.

For example, you want to provide a 2-factor authentication for managers to post some forms. Then another configuration "2factorAuth" can be defined for "mobileApp" application with ReadRoles set to "Manager". Then, the whole application storage for "mobileApp" looks like:

Sample Code

```
{
  "ApplicationId": "mobileApp",
  "HomeScreen": {
    "Menu": {
      "AutoHiding": false,
      "Style": "blue",
      "AutoHiding@odata.type": "Edm.Boolean",
      "Style@odata.type": "Edm.String"
    },
    "ShortCuts": ["Overview", "Profile"],
    "__metadata": {
      "ReadRoles": [],
      "WriteRoles": ["Administrator"] },
    "Menu@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
    "ShortCuts@odata.type": "Collection(Edm.String)",
    "__metadata@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageMetadata"
  },
  "2factorAuth": {
    "RequestToken": {
      "url": "https://2factor.server/requesttoken/phone",
      "method": "POST",
      "parameters": ["username", "phone"]
    },
    "VerifyToken": {
      "url": "https://2factor.server/verify",
      "method": "POST",
      "parameters": ["token"]
    },
    "__metadata": {
      "ReadRoles": ["Manager"],
      "WriteRoles": ["Administrator"] },
    "RequestToken@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
    "VerifyToken@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
    "__metadata@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageMetadata"
  },
  "HomeScreen@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
  "2factorAuth@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject"
}
```

Then, the `https://<mobile services host>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/2factorAuth?overwrite=true` can return settings according to the login user's role, and mobile app can show screens and control flows according to the 2factorAuth settings.

The Application Storage configuration can also be used to persist settings for server application, if both ReadRoles and WriteRoles are defined as a role (or roles) which never belong to any users. Then, your server application can read/write these application storage configuration using a technical user who has the role.

Storage service supports two type of APIs:

- Admin APIs
- Runtime APIs

[Runtime APIs \[page 451\]](#)

Used by mobile application to access settings at all the three storages level during runtime.

[Admin APIs \[page 465\]](#)

Admin APIs are used to configure settings at application storage level, and manages all settings in both user storage and device storage.

Related Information

[Storage Service \[page 393\]](#)

[Runtime APIs \[page 451\]](#)

[Admin APIs \[page 465\]](#)

1.6.1.3.16.1 Runtime APIs

Used by mobile application to access settings at all the three storages level during runtime.

The runtime API can be accessed by both application authentication and origin authentication URL patterns. Currently, the runtime APIs only addressed origin authentication URL pattern. For example,

- Origin authentication URL pattern: `https://<mobile services host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>[?overwrite=<overwrite>]`
- Application authentication URL pattern: `https://<mobile services host>/mobileservices/application/<applicationId>/<serviceName>/...` Request is authenticated by application's security configuration (configured in Mobile Services cockpit).

If mobile application only needs to access default settings, it can directly use application storage level API and retrieve the settings. If mobile application allows their end users to create user or device specific settings, it uses user storage or device storage APIs. If there are no user level or device level settings, then the GET operation response includes only application level settings. The device storage API merges settings in both Application storage and user storage.

HTTP Method	Action
GET	GET Storage Service [page 452]
POST	POST Storage Service [page 455]

HTTP Method	Action
<i>PUT</i>	PUT Storage Service [page 458]
<i>DELETE</i>	DELETE Storage Service [page 463]
<i>PATCH</i>	PATCH Storage Service [page 461]

[GET Storage Service \[page 452\]](#)

Retrieve a mobile data storage service based on application configuration, user property, or device property.

[POST Storage Service \[page 455\]](#)

Create a mobile data storage service based on application configuration, user property, or device property.

[PUT Storage Service \[page 458\]](#)

Update mobile data storage service based on application configuration, user property, or device property.

[PATCH Storage Service \[page 461\]](#)

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

[DELETE Storage Service \[page 463\]](#)

Delete a mobile data storage service based on application configuration, user property, or device property.

1.6.1.3.16.1.1 GET Storage Service

Retrieve a mobile data storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

HTTP Method *GET*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application identifier.	String

Parameter	Type	Description	Parameter Type
OriginName	Required	Origin from where the application is created. The valid value is "hcpms".	String
configurationName	Optional	Configuration name.	String
propertyName	Optional	Property name. The propertyName, propertyNameLv2 or propertyNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of the properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.	String

Note

Any special characters in the configuration name and property name includes must be correctly encoded in URL.

Request Example

```
GET /application/appID1/endpoints/__metadata/readRoles
- get the read roles of endpoints configuration.
GET /application/appID1/endpoints/epl/url
- get the url value of epl endpoint.
```

Response Status and Error Codes

Code	Description
200	Property value, which may be a JSON object, a primitive value, or any array except for JSON one.
403	No permission to access a configuration.
404	Specified application or property not exists.
500	Unknown error.

Request (By User Name)

Retrieve user-specific configuration.

URI:`https://<mobile_services_host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
user name	Required	User name	String
Overwrite	Optional	Whether result needs to merge with application-level data. The default value is true.	Boolean

Response Status and Error Codes

Code	Description
200	User property created.
403	No permission to access a configuration.
404	Specified application or property does not exists.
500	Unknown error.

Note

Set the overwrite property for a user-level configuration to true, to overwrite an application-level configuration. The current level and its parent level configuration is merged by using the overwrite property.

Request (By Device Name)

Retrieve a device-specific configuration or property.

URI:`https://<mobile_services_host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/device /<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Unique identifier for device type	String
Overwrite	Optional	Whether result needs to merge with application-level data. The default value is true.	Boolean

Code	Description
200	Device property created.
403	No permission to access a configuration.
404	Device property already exists.
500	Unknown error.

Related Information

[Storage Service \[page 393\]](#)

1.6.1.3.16.1.2 POST Storage Service

Create a mobile data storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

Create an application configuration, which is a JSON object that includes arbitrary properties. Each configuration can include a "__metadata" property which includes two properties for access control: readRoles and writeRoles.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

HTTP Method *POST*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application identifier	String
configurationName	Optional	Configuration name	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String

Parameter	Type	Description	Parameter Type
propName	Optional	Property name. The propName, propNameLv2, or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. The property name must exists in the request URL, otherwise, a 404 code for PUT/PATCH/DELETE/GET methods is returned.	String

Request Example

This example code creates a configuration for an application named "test1".

```
POST Storage/v1/runtime/application/test1
{
  "endpoints": {
    "ep1": { "url": "http://endpoint1" },
    "ep2": { "url": "http://endpoint2" }
  },
  "push": {
    "APN": { "host": "www.apple.com",
              "user": "useraaa",
              "pwd": "pwd" },
    "GCM": { "host": "www.google.com",
              "token": "ABABABA" },
    "__metadata": {
      "readRoles": ["Role1", "Role2"],
      "writeRoles": ["Role1"] }
  }
}
```

After creating the application configuration, you can use property names to add a new endpoint for the application.

```
{
  "endpoints": {
    "ep1": { "url": "http://endpoint1" },
    "ep2": { "url": "http://endpoint2" },
    "ep3": { "url": "http://endpoint3" },
  },
  "push": {
    "APN": { "host": "www.apple.com",
              "user": "useraaa",
              "pwd": "pwd" },
    "GCM": { "host": "www.google.com",
              "token": "ABABABA" },
    "__metadata": {
      "readRoles": ["Role1", "Role2"],
      "writeRoles": ["Role1"] }
  }
}
```


Response Status and Error Codes

Code	Description
201	Property created.
403	No permission to access a configuration.
409	Property already exists.
500	Unknown error.

Request (By User Name)

Create a user-specific configuration.

URI: `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
user name	Required	User name	String

Request Example

```
POST /application/appID1/user/ab01/ep1
{ "authenticationType": "NoAuth" }
```

Response Status and Error Codes

Code	Description
201	Property created.
403	Cannot access other user configuration.
409	Property already exists.
500	Unknown error.

Note

The user name must be the authenticated user name, otherwise, the service returns error code 403.

Request (By Device Name)

Create a device-specific configuration or property.

Details about the creator of the device configuration is stored internally. The storage service API verifies the creator information before (??) accessing device configuration.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Unique identifier for device type	String

Code	Description
201	Device created.
403	No permission to access a configuration.
409	Property already exists.
500	Unknown error.

Note

The authenticated user name implicitly accesses the information; however, the user name and device ID identify the device-level configuration.

1.6.1.3.16.1.3 PUT Storage Service

Update mobile data storage service based on application configuration, user property, or device property.

Request (By application configuration)

URI `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

HTTP Method *PUT*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Mandatory	Application identifier.	String

Parameter	Type	Description	Parameter Type
originName	Mandatory	Origin from where the application is created. The valid value is "hcpms".	String
configurationName	Optional	Configuration name.	String
propertyName	Optional	Property name. The propertyName, propertyNameLv2 or propertyNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.	String

Note

It replaces the stored configurations by the one provided in request body.

Response Status and Error Codes

Code	Description
204	Updated application configuration successfully.
403	No permission to access a configuration.
409	Specified application or property not exists.
500	Unknown error.

Request (By User Name)

Update a user-specific configuration or property.

URI: `https://<mobile services host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propertyName>/<propertyNameLv2>/<propertyNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
userName	Mandatory	User name	String

Response Status and Error Codes

Code	Description
204	Updated user property successfully.
403	Cannot access other user's configuration.
404	Specified application or property does not exists.
500	Unknown error.

Request (By Device Name)

Create a device-specific configuration or property.

Details about the creator of the device configuration is stored internally. The storage service API verifies the creator information before (??) accessing device configuration.

URI:`https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device /<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Unique identifier for device type	String

Code	Description
201	Device created.
403	No permission to access a configuration.
409	Property already exists.
500	Unknown error.

Note

The authenticated user name implicitly accesses the information; however, the user name and device ID identify the device-level configuration.

Related Information

[Storage Service \[page 393\]](#)

1.6.1.3.16.1.4 PATCH Storage Service

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

The partial configuration after merged with the existing configuration is stored in the database.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

HTTP Method *PATCH*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application identifier.	String
prop	Required	Partial property value, which can be part of a JSON object, a primitive value, or any array except a JSON array.	Property
originName	Mandatory	Origin from where the application is created. The valid value is "hcpms".	String
configurationName	Optional	Configuration name.	String
propName	Optional	Property name. The propName, propNameLv2 or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If the property does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.	String

Request Example

In this example, the following code is in application configuration 'applD1'

```
{  
  "endpoints": {
```

```

    "ep2": { "url": "http://newEndpoint2" }
  },
  "push": {
    "BES": { "host": "www.blackberry.com",
              "user": "useraaa",
              "pwd": "pwd" }
  }
}

```

After existing and new configuration is merged, the new configuration is stored in the database as:

```

{
  "endpoints": {
    "ep1": { "url": "http://endpoint1" },
    "ep2": { "url": "http://newEndpoint2" }
  },
  "push": {
    "APN": { "host": "www.apple.com",
              "user": "useraaa",
              "pwd": "pwd" },
    "GCM": { "host": "www.google.com",
              "token": "ABABABA" },
    "BES": { "host": "www.blackberry.com",
              "user": "useraaa",
              "pwd": "pwd" },
    "__metadata": {
      "readRoles": [ "Role1", "Role2" ],
      "writeRoles": [ "Role1" ]
    }
  }
}

```

Response Status and Error Codes

Code	Description
204	Updated application configuration or property.
403	No permission to access a configuration.
404	Specified application configuration does not exist. .
500	Unknown error.

Request (By User Name)

Partially update a user-specific configuration or property.

URI: `https://<mobile services host>/`

`mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
userName	Required	User name	String

Response Status and Error Codes

Code	Description
204	Updated user property.
403	Cannot access other user's configuration.
409	Property already exists.
500	Unknown error

Request (By Device Name)

Partially update a device-specific configuration or property.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Unique identifier for device type	String

Code	Description
204	Updated device property.
403	No permission to access a configuration.
409	Property already exists.
500	Unknown error.

1.6.1.3.16.1.5 DELETE Storage Service

Delete a mobile data storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

Delete application configuration or property.

URI `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

HTTP Method *DELETE*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Mandatory	Application identifier	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String
configurationName	Optional	Configuration name	String
propName	Optional	Property name. The propName, propNameLv2 or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.	String

Response Status and Error Codes

Code	Description
204	Deleted of application configuration or property.
403	No permission to access a configuration.
404	Specified application configuration does not exists.
500	Unknown error.

Request (By User Name)

Delete a user-specific configuration or property.

URI:`https://<mobile services host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
userName	Mandatory	User name	String

Response Status and Error Codes

Code	Description
204	Deleted user property successfully.
403	Cannot access other user's configuration.
404	User property already exists.
500	Unknown error

Request (By Device Name)

Delete a device-specific configuration or property.

URI:`https://<mobile services host>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/device /<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Mandatory	Unique identifier for device type	String

Response Status and Error Codes

Code	Description
204	Updated device property.
403	No permission to access a configuration.
404	Specified application or device property already exists.
500	Unknown error.

1.6.1.3.16.2 Admin APIs

Admin APIs are used to configure settings at application storage level, and manages all settings in both user storage and device storage.

The Admin API can only be accessed by origin authentication URL pattern. It supports two authentication types: Basic, App2App.

Note

Currently, only SAP Mobile Services originated applications are supported.

HTTP Method	Action
<i>GET</i>	GET Storage Services [page 467]
<i>POST</i>	POST Storage Services [page 466]
<i>PUT</i>	PUT Storage Services [page 470]
<i>DELETE</i>	DELETE Storage Services [page 473]
<i>PATCH</i>	PATCH Storage Services [page 471]

[POST Storage Services \[page 466\]](#)

Add new entity to application, user, and device storage sets.

[GET Storage Services \[page 467\]](#)

Retrieve new entities from application, user, and device storage sets.

[PUT Storage Services \[page 470\]](#)

Update entities from application, user, and device storage sets.

[PATCH Storage Services \[page 471\]](#)

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

[DELETE Storage Services \[page 473\]](#)

Delete entities from application, user, and device storage sets.

1.6.1.3.16.2.1 POST Storage Services

Add new entity to application, user, and device storage sets.

Request (By Application Storage Set)

Create a new entity to ApplicationStorageSet

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet`

HTTP Method *POST*

Response Status and Error Codes

Code	Description
201	Entity created.
409	Entity already exists.
500	Unknown error.

Request (By User Storage Set)

Create a new entity in user storage set.

URI:`https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet`

Response Status and Error Codes

Code	Description
201	Entity created.
409	Entity already exists.
500	Unexpected error.

Request (By Device Storage Set)

Create a new entity in device storage set.

URI:`https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet`

Response Status and Error Codes

Code	Description
201	Device entity created.
409	Entity already exists.
500	Unexpected error.

1.6.1.3.16.2.2 GET Storage Services

Retrieve new entities from application, user, and device storage sets.

Request (By Application Storage Set)

Retrieve an entity from application storage set by key.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet/(ApplicationId='<ApplicationId>')`

Retrieve all the entities from application storage.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet?${<system_query_option>}`

Note

Supported OData system query options are \$count, \$skip, \$top.

HTTP Method **GET**

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Key: application ID	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String
\$top	Optional	Show only the first n elements.	Integer
\$skip	Optional	Skip the first n elements.	Integer
\$count	Optional	Allows users to request a count of the matching resources. The number is included with the resources in the response.	Boolean

Response Status and Error Codes

Code	Description
200	Entity set found in application storage set.
500	Unexpected error.
404	Entity does not exist

Request (By User Storage Set)

Retrieve an entity from user storage set by key.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId='<ApplicationId>', Username= '<username>')`

Retrieve all entities from user storage set.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet?$<system_query_option>`

Request Parameters

Parameter	Type	Description	Parameter Type
username	Required	User name	String

Response Status and Error Codes

Code	Description
200	Entity set created.
404	Entity does not exist.
500	Unknown error.

Request (By Device Storage Set)

Retrieve a entity from device storage set by key.

URI:`https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId='<ApplicationId>', Username= '<username>', DeviceId='<deviceId>')`

Retrieve all entities from device storage set.

URI:`https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet?$<system_query_option>`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Unique identifier for device type	String

Response Status and Error Codes

Code	Description
200	Entity set created.
404	Device property already exists.
500	Unexpected error.

1.6.1.3.16.2.3 PUT Storage Services

Update entities from application, user, and device storage sets.

Request (By Application Storage Set)

Update entity in application storage set.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet(' <ApplicationId>')`

HTTP Method *PUT*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	application ID	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String

Response Status and Error Codes

Code	Description
200	Entity updated in application storage set.
500	Unexpected error.
404	Entity does not exist

Request (By User Storage Set)

Update entities in user storage set.

URI:`https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId=' <ApplicationId>', Username= '<username>')`

Request Parameters

Parameter	Type	Description	Parameter Type
username	Required	User name	String

Response Status and Error Codes

Code	Description
200	Entity set updated in user storage set.
404	Entity does not exist.
500	Unexpected error.

Request (By Device Storage Set)

Update entity in device storage set.

URI: `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId='<ApplicationId>', Username= '<username>', DeviceId='<deviceId>')`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Device ID	String

Response Status and Error Codes

Code	Description
200	Entity updated in device storage set.
404	Entity does not exist.
500	Unexpected error.

1.6.1.3.16.2.4 PATCH Storage Services

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

Request (By Application Storage Set)

Update entity in application storage set.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet(<ApplicationId>')`

HTTP Method *PATCH*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	application ID	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String

Response Status and Error Codes

Code	Description
200	Entity updated in application storage set.
500	Unexpected error.
404	Entity does not exist

Request (By User Storage Set)

Update entities in user storage set.

URI:`https://<mobile services host>/mobileservices/origin/<originname>/Storage/v1/admin/UserStorageSet(ApplicationId='<ApplicationId>', Username= '<username>')`

Request Parameters

Parameter	Type	Description	Parameter Type
username	Required	User name	String

Response Status and Error Codes

Code	Description
200	Entity set is updated in the user storage set.
404	Entity does not exist.
500	Unexpected error.

Request (By Device Storage Set)

Update entity in device storage set.

URI:`https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId='<ApplicationId>', Username= '<username>', DeviceId='<deviceId>')`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Device ID	String

Response Status and Error Codes

Code	Description
200	Entity set is updated in the device storage set.
404	Entity does not exist.
500	Unexpected error.

1.6.1.3.16.2.5 DELETE Storage Services

Delete entities from application, user, and device storage sets.

Request (By Application Storage Set)

Delete entity from application storage set.

URI `https://<mobile services host>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet(' <ApplicationId>')`

HTTP Method *DELETE*

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	application ID	String
originName	Required	Origin from where the application is created. The valid value is "hcpms".	String

Response Status and Error Codes

Code	Description
204	Entity deleted from application storage set.
500	Unexpected error.
404	Entity does not exist.

Request (By User Storage Set)

Delete entities from user storage set.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId='<ApplicationId>', Username= '<username>')`

Request Parameters

Parameter	Type	Description	Parameter Type
username	Required	User name	String

Response Status and Error Codes

Code	Description
204	Entity set is deleted from the user storage set.
404	Entity does not exist.
500	Unexpected error.

Request (By Device Storage Set)

Delete entities from device storage set.

URI: `https://<mobile_services_host>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId='<ApplicationId>', Username= '<username>', DeviceId='<deviceId>')`

Request Parameters

Parameter	Type	Description	Parameter Type
deviceId	Required	Device ID	String

Response Status and Error Codes

Code	Description
204	Entity set is deleted in the device storage set.
404	Entity does not exist.
500	Unexpected error.

1.6.1.3.17 Client Resources Service

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Mobile Services Admin Cockpit.

Client Resources service supports two types of APIs:

- Runtime APIs
- Admin APIs

[Runtime APIs \[page 475\]](#)

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

[Admin APIs \[page 478\]](#)

The Client Resource Bundle API allows the application to download the resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

Related Information

[Runtime APIs \[page 475\]](#)

[Admin APIs \[page 478\]](#)

1.6.1.3.17.1 Runtime APIs

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

Runtime API

HTTP Method	Action
GET	GET Client Resources Service [page 475]

[GET Client Resources Service \[page 475\]](#)

1.6.1.3.17.1.1 GET Client Resources Service

HTTP Method: [GET](#)

Default Resource Bundle

This method allows you to download the default resource bundle an application.

URL: `https://<mobile_services_host>/mobileservices/application/{applicationId}/bundles/v1/runtime/bundle/application/{applicationId}`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application ID	string

Response Status and Error Codes

Code	Description
200	Resource bundle downloaded successfully. X-BUNDLE-EXTENSION - The file extension as a response header.
404	Application or resource bundle not found.
500	Unknown error.

Specified Resource Bundle

This method allows you to download the specified resource bundle of an application.

URL: `https://<mobile_services_host>/mobileservices/application/{applicationId}/bundles/v1/runtime/bundle/application/{applicationId}/bundle/{bundlename}/version/{version}`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application ID	string
bundlename	Required	Bundle name in the format bundlename:version.	string
version	Required	Bundle version in the format bundlename:version.	string

Response Status and Error Codes

Code	Description
200	Resource bundle downloaded successfully. X-BUNDLE-EXTENSION - The file extension as a response header.
404	Application or resource bundle not found.
500	Unknown error.

List All Resource Bundles

This method allows you to list all the resource bundles of an application.

URL: `https://<mobile services host>/mobileservices/application/{application}/bundles/v1/runtime/bundle/application/{applicationId}/list`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application ID.	string

Response Status and Error Codes

Code	Description
200	Resource bundles listed successfully.
404	Application not found.
500	Unknown error.

Downloaded the Latest Resource Bundle

This method allows you to download the latest resource bundle of an application by ordering bundle versions lexicographically.

❖ Example

- 1.1.1.2 is a later version than 1.1.1
- 1.1.1.2 is a later version than 1.1.1.1.a
- 1.2 a.1 is a later version than 1.1.1.1.a
- 1.a is a later version than 1.2.a.1

URL: `https://<mobile services host>/mobileservices/application/{application}/bundles/v1/runtime/bundle/application/{applicationId}/bundle/{bundlename}`

Request Parameters

Parameter	Type	Description	Parameter Type
If-None-Match	Optional	The etag value returned during the last GET latest resource bundle request	string
applicationId	Required	Application ID	string
bundlename	Required	Bundle name	string

Response Status and Error Codes

Code	Description
200	Resource bundle downloaded successfully: <ul style="list-style-type: none">• etag - etag value of downloaded resource bundle.• X-BUNDLE-EXTENSION - the file extension as a response header.
304	Latest resource bundle of an application.
404	Application or resource bundle not found.
500	Unknown error.

1.6.1.3.17.2 Admin APIs

The Client Resource Bundle API allows the application to download the resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

Admin API

HTTP Method	Action
GET	GET Client Resources Service [page 478]
POST	POST Client Resources Service [page 481]
DELETE	DELETE Client Resources Service [page 482]
PUT	PUT Client Resources Service [page 483]

[GET Client Resources Service \[page 478\]](#)

[POST Client Resources Service \[page 481\]](#)

[DELETE Client Resources Service \[page 482\]](#)

This method allows you to delete an entity from the Application Resource Bundles by using the key.

[PUT Client Resources Service \[page 483\]](#)

This method allows you to set the binary content of the application resource.

1.6.1.3.17.2.1 GET Client Resources Service

HTTP Method: [GET](#)

Service Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin`

Response Status and Error Codes

Code	Description
200	Service document.
500	Unexpected error.

Metadata Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata`

Response Status and Error Codes

Code	Description
200	Metadata document.
500	Unexpected error.

Application Resource Bundles

This method allows you to get the entities from the Entity Set Application Resource Bundles.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles`

Response Status and Error Codes

Code	Description
200	EntitySet ApplicationResourceBundles.
default	Unexpected error.

Application Resource Bundles by Key

This method allows you to get the entity from the Application Resource Bundles by key, and it returns the entity with the key from the Application Resource Bundles.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationId	Required	key: ApplicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Parameter	Type	Description	Parameter Type
BundleName	Required	key: BundleName	string
BundleVersion	Required	key: BundleVersion	string

Response Status and Error Codes

Code	Description
200	EntitySet ApplicationResourceBundles.
404	Entity Not Found.
default	Unexpected error.

Binary Content of the Application Resource

This method allows you to get the binary content of the application resource

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/BundleFile`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationId	Required	key: ApplicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string
BundleName	Required	key: BundleName	string
BundleVersion	Required	key: BundleVersion	string

Response Status and Error Codes

Code	Description
200	The binary content of the application resource.
404	Entity Not Found.
default	Unexpected error.

1.6.1.3.17.2.2 POST Client Resources Service

HTTP Method: *POST*

Application Resource Bundles

This method allows you to add a new entity to the Entity Set Application Resource Bundles. The uploaded logs can be viewed from SAP Mobile Services Admin Cockpit.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationResourceBundle	Optional	The new entity.	<ul style="list-style-type: none">ApplicationId: stringBundleName: stringBundleVersion: stringBundleExtension: stringIsDefault: boolean
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Response Status and Error Codes

Code	Description
201	Created entity.
409	Entity already exists.
default	Unexpected error.

Set Default

This method allows you to invoke the action Set Default.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/com.sap.mobile.server.bundle.admin.v1.SetDefault`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationId	Required	key: ApplicationId	string

Parameter	Type	Description	Parameter Type
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string
BundleName	Required	key: BundleName	string
BundleVersion	Required	key: BundleVersion	string

Response Status and Error Codes

Code	Description
200	Empty response.
404	Entity Not Found.
default	Unexpected error.

1.6.1.3.17.2.3 DELETE Client Resources Service

This method allows you to delete an entity from the Application Resource Bundles by using the key.

HTTP Method: [DELETE](#)

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationId	Required	key: ApplicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string
BundleName	Required	key: BundleName	string
BundleVersion	Required	key: BundleVersion	string

Response Status and Error Codes

Code	Description
200	EntitySet ApplicationResourceBundles.
404	Entity Not Found.
default	Unexpected error.

1.6.1.3.17.2.4 PUT Client Resources Service

This method allows you to set the binary content of the application resource.

HTTP Method: *PUT*

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/BundleFile`

Request Parameters

Parameter	Type	Description	Parameter Type
ApplicationId	Required	key: ApplicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string
BundleName	Required	key: BundleName	string
BundleVersion	Required	key: BundleVersion	string
resourceBinary	Optional	The binary content of the application resource.	binary

Response Status and Error Codes

Code	Description
204	Set the binary content of the application resource successfully.
404	Entity Not Found.
default	Unexpected error.

1.6.1.3.18 Client Log Upload Service

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

Client Log Upload service supports two types of APIs:

- Runtime APIs
- Admin APIs

[Runtime APIs \[page 484\]](#)

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

[Admin APIs \[page 486\]](#)

The Admin API is used for managing the Log Upload policy. SAP Mobile Services logging Admin API provides a unique interface for all the logs, and it allows you to get the uploaded client logs.

Related Information

[Runtime APIs \[page 484\]](#)

[Admin APIs \[page 486\]](#)

1.6.1.3.18.1 Runtime APIs

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

Runtime API

HTTP Method	Action
<i>POST</i>	POST Client Log Upload Service [page 485]

[POST Client Log Upload Service \[page 485\]](#)

The Client Log Upload API allows the application to upload the client logs to the server for further analysis. This method allows you to upload client logs for an application. If the client log upload option is enabled for the specified application, then the client logs get uploaded.

1.6.1.3.18.1.1 POST Client Log Upload Service

The Client Log Upload API allows the application to upload the client logs to the server for further analysis. This method allows you to upload client logs for an application. If the client log upload option is enabled for the specified application, then the client logs get uploaded.

Note

Anyone with access to Mobile Services cockpit can see the uploaded logs.

HTTP Method: *POST*.

URL: `https://<mobile services host>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	Application identifier	String
deviceId	Optional	Device identifier. Its value will be saved with uploaded log if specified.	String
clientlogs	Required	Client logs	Client logs <any>Serialized Client logs. It can be zipped data, multipart data or simple binary data. Log format: ~~~#Date time#Severity#CorrelationId#Source#Location#Message# ~~~ The format of Date time is "YYYY-MM-DDThh:mm:ss.sTZD", such as "2014-07-01T17:16:08.637+02:00". The valid values of Severity are DEBUG, INFO, WARN, ERROR and FATAL. The CorrelationId value is optional, and can be left as empty if client cannot provide it for each log message. The Message value is log message generated by client, and # character in it must be escaped to avoid corrupt uploading.

Response Status and Error Codes

Code	Description
204	Client logs uploaded.
400	Invalid log format.
403	Client log upload is not enabled for the application.
404	No application found.
500	Unexpected error.

1.6.1.3.18.2 Admin APIs

The Admin API is used for managing the Log Upload policy. SAP Mobile Services logging Admin API provides a unique interface for all the logs, and it allows you to get the uploaded client logs.

Admin API

HTTP Method	Action
GET	GET Client Log Upload Service [page 486]
POST	POST Client Log Upload Service [page 488]
PATCH	PATCH Client Log Upload Service [page 488]
DELETE	DELETE Client Log Upload Service [page 489]

[GET Client Log Upload Service \[page 486\]](#)

[POST Client Log Upload Service \[page 488\]](#)

This method allows you to post the new entity to the Entity Set Application Policy.

[PATCH Client Log Upload Service \[page 488\]](#)

This method allows you to update the entity in the Entity Set Application Policy.

[DELETE Client Log Upload Service \[page 489\]](#)

This method allows you to delete an entity in the Entity Set Application Policy.

1.6.1.3.18.2.1 GET Client Log Upload Service

HTTP Method: [GET](#)

Service Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin`

Response Status and Error Codes

Code	Description
200	Service document.
500	Unexpected error.

Metadata Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/$metadata`

Response Status and Error Codes

Code	Description
200	Metadata document.
500	Unexpected error.

Entity Set Application Policy

This method allows you to get the entities from the Entity Set Application Policy.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet`

Response Status and Error Codes

Code	Description
200	EntitySet ApplicationPolicy
404	Not found.
500	Unexpected error.

Application Policy by Key

This method allows you to get the entity from the Application Policy by key, and it returns the entity with the key from the Application Policy.

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Response Status and Error Codes

Code	Description
200	EntitySet ApplicationPolicy.
404	Not found.
500	Unexpected error.

1.6.1.3.18.2.2 POST Client Log Upload Service

This method allows you to post the new entity to the Entity Set Application Policy.

HTTP Method: *POST*

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet`

Parameter	Type	Description	Parameter Type
ApplicationPolicy	Required	The new entity	<ul style="list-style-type: none">ApplicationId: stringLogUploadEnabled: boolean
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Response Status and Error Codes

Code	Description
201	Created entity.
409	Client log upload policy already exists.
500	Unexpected error.

1.6.1.3.18.2.3 PATCH Client Log Upload Service

This method allows you to update the entity in the Entity Set Application Policy.

HTTP Method: *PATCH*

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	string
ApplicationPolicy	Required	The entity to patch.	<ul style="list-style-type: none"> ApplicationId: string LogUploadEnabled: boolean
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Response Status and Error Codes

Code	Description
204	Empty response.
404	Not Found.
500	Unexpected error.

1.6.1.3.18.2.4 DELETE Client Log Upload Service

This method allows you to delete an entity in the Entity Set Application Policy.

HTTP Method: *DELETE*

URL: `https://<mobile services host>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	string
Origin	Required	Origin from where the application is created. The valid value is "hcpms".	string

Response Status and Error Codes

Code	Description
204	Empty response.
404	Not found.
500	Unexpected error.

1.6.1.3.19 Push as API Service

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

Developers enable native push notification in the application code, and link the corresponding certificate with the mobile application at build time. Users download the application from a market place, such as Apple App Store, Google Play, or similar service, and, when a change occurs in the back end, a push notification is sent to mobile applications on devices that have push enabled.

The Push as API Service also allows you to send SMS notifications. You can send an SMS notification to the application instead of a native push notification. The Push as API service allows you to use both SMS notifications as well as native push notifications.

The Push as API service supports the following types of APIs:

- Runtime APIs
- Admin APIs
- Back-end APIs

[Runtime APIs \[page 490\]](#)

The mobile services API can be consumed by mobile applications running on a device, server applications hosted on HCP, and any other back-end applications that can access the API over the internet.

[Admin APIs \[page 498\]](#)

The Administrator role is required to access Admin APIs.

[Back-end APIs \[page 503\]](#)

The notification user role is required to access back-end APIs.

Related Information

[Runtime APIs \[page 490\]](#)

[Admin APIs \[page 498\]](#)

[Back-end APIs \[page 503\]](#)

1.6.1.3.19.1 Runtime APIs

The mobile services API can be consumed by mobile applications running on a device, server applications hosted on HCP, and any other back-end applications that can access the API over the internet.

The Push as API service provides services for mobile devices to register for the push service of a specific application in order to receive notifications, as well as the services to manage the lifecycle of their registration. When registering a device it is now possible to specify a phone number (MSISDN) in the register device request payload:

```
{
```

```

"deviceModel": "Android",
"msisdn": "+491234567890"
}

```

- Make sure that the phone number is in an international format (+, country code, network code, number), and is stored along the device registration.
- You can register a device for both SMS notification and native push.
- Example of registering a device for SMS notification and GCM:

❖ Example

```

{
  "deviceModel": "Android",
  "pushToken": "abcdefghijklmnopqrstuvxyz1234567890",
  "msisdn": "+49123456789"
}

```

- Example of registering a device for Baidu:

❖ Example

```

{
  "deviceModel": "Android",
  "pushToken": "<channelid>",
  "msisdn": "+49123456789"
}

```

When you use JSON Web Token (JWT) authentication, the `sap.mob.roles` element containing the role should be a part of the JWT. For example, to access a runtime service, the JWT payload is `{"sub": "P1940703319", "sap.mob.roles": ["Notification User"], "iss": "hcpms"}`.

Runtime API

HTTP Method	Action
GET	GET Push as API Service [page 492]
POST	POST Push as API Service [page 492]
PUT	PUT Push as API Service [page 494]
PATCH	PATCH Push as API Service [page 497]
DELETE	DELETE Push as API Service [page 498]

[GET Push as API Service \[page 492\]](#)

Enables a device to retrieve an application ID or a sender ID. For example, you could use the GET method to query for the Google Cloud Messaging (GCM) sender ID before registering a device.

[POST Push as API Service \[page 492\]](#)

Uploads push tokens to SAP Mobile Services.

[PUT Push as API Service \[page 494\]](#)

Updates an existing push token.

[PATCH Push as API Service \[page 497\]](#)

This method allows you to merge push tokens. If no device registration exists for the given device ID, the registration is created on the fly.

[DELETE Push as API Service \[page 498\]](#)

Delete a native push configuration.

1.6.1.3.19.1.1 GET Push as API Service

Enables a device to retrieve an application ID or a sender ID. For example, you could use the GET method to query for the Google Cloud Messaging (GCM) sender ID before registering a device.

URI:`https://<mobile_services_host>/mobileservices/push/v1/runtime/applications/{applicationId}/pushconfigurations/os/{operatingSystem}/pushid`

HTTP Method [GET](#)

Request Parameters

Parameter	Description	Required	Parameter Type
<code>applicationId</code>	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String
<code>operatingSystem</code>	Device Type	Yes	String; (windows android)

Response Status and Error Codes

Code	Description
200	Push ID returned.
400	Operating system not supported.
404	Application not found.
500	Cannot load push sender ID.

1.6.1.3.19.1.2 POST Push as API Service

Uploads push tokens to SAP Mobile Services.

The Push endpoint lets you register push tokens, which deliver native push notifications to devices. Devices must indicate the platform they run on, to enable connection to the correct notification delivery service.

ⓘ Note

This service overrides any existing device registration push data information. The existing values, which are not set in the `updateDeviceRegistrationPushData` element, are reset.

Request (by Device)

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
registerDevicePushData	Enables native push delivery.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String
operatingSystem	Device Type	Yes	String; (ios windows android baidu)

Response Status and Error Codes

Code	Description
201	Registration successful.
400	Incomplete request.
409	Device ID already exists for the given application.
500	Registration failed.

Request (by Device ID)

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
registerDeviceByIdPushData	Enables native push delivery.	Yes	String

Parameter	Description	Required	Parameter Type
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String
operatingSystem	Device Type	Yes	String; (ios windows android baidu)
deviceId	Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.	Yes	String

Response Status and Error Codes

Code	Description
201	Registration successful.
400	Incomplete request.
409	Device ID already exists for the given application.
500	Registration failed.

1.6.1.3.19.1.3 PUT Push as API Service

Updates an existing push token.

Request (by Device ID)

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}`

HTTP Method *PUT*

Request Parameters

Parameter	Description	Required	Parameter Type
updateDeviceRegistrationPushData	Enables native push delivery.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String
operatingSystem	Device Type	Yes	String; (ios windows android baidu)
deviceId	Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.	Yes	String

Response Status and Error Codes

Code	Description
200	Registration update successful.
400	The request does not contain all required information.
404	Registration not found for the given Device ID.
500	Registration update failed.

Updates the status of an existing notification. This service can be called by connected client devices and two status' - received and consumed are allowed.

Request (by Application ID)

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/applications/{applicationId}/notifications/{notificationId}/status/{status}`

Updates the status of an existing notification. Connected client devices are allowed to call only the received or consumed status respectively.

HTTP Method *PUT*

Request Parameters

Parameter	Description	Required	Parameter Type
applicationId	An application ID is required for authentication.	Yes	String
notificationId	The ID of the notification for which the status has to be updated.	Yes	String
status	The new status which needs to be set, only received and consumed are allowed.	Yes	String; (received consumed)

Response Status and Error Codes

Code	Description
200	Registration update successful.
400	The request does not contain all required information.
404	Notification not found for the given ID.

Request (by Notification ID)

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/notifications/{notificationId}/status/{status}`

HTTP Method *PUT*

Request Parameters

Parameter	Description	Required	Parameter Type
notificationId	The ID of the notification for which the status has to be updated.	Yes	String
status	The new status which needs to be set, only received and consumed are allowed.	Yes	String; (received consumed)

Response Status and Error Codes

Code	Description
200	Registration update successful.

Code	Description
400	The request does not contain all required information.
404	Notification not found for the given ID.

1.6.1.3.19.1.4 PATCH Push as API Service

This method allows you to merge push tokens. If no device registration exists for the given device ID, the registration is created on the fly.

URI: `https://<mobile services host>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}`

HTTP Method [PATCH](#)

Request Parameters

Parameter	Description	Required	Parameter Type
mergeDeviceRegistrationPushData	Enables native push delivery.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String
operatingSystem	Device Type	Yes	String; (ios windows android baidu)
deviceId	Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.	Yes	String

Response Status and Error Codes

Code	Description
200	Registration merged.
400	The request does not contain all required information.
500	Registration update failed.

1.6.1.3.19.1.5 DELETE Push as API Service

Delete a native push configuration.

URI: `https://<mobile services host>mobileservices/origin/hcpms/push/v1/admin/applications/{applicationId}/pushconfigurations`

HTTP Method *DELETE*

Request Parameters

Parameter	Description	Required	Parameter Type
applicationId	The ID of the current application. This should match the application ID used while configuring push on HCPms for your app.	Yes	String

Response Status and Error Codes

Code	Description
204	Push configuration successfully deleted.
400	Invalid request.
403	Access denied for the application.
404	Application not found.
500	Push configuration deletion failed.

1.6.1.3.19.2 Admin APIs

The Administrator role is required to access Admin APIs.

Push as API service provides services to manage the push configurations on at the application level. It also supports all the major push providers, such as APNs, GCM, WNS, and it can be configured individually. The existing REST and OData Admin services supports SMS, which integrates all the relevant configuration options. You can configure the SMS notification similar to the native push configuration. An example request payload to configure an application for SMS looks like the following

❖ Example

```
{
  "gcm": {
    "apiKey": "12341234",
    "senderId": "abc"
  },
  "sms": {
    "login": "userLogin",
    "password": "blabla",
    "originatingAddress": "TheOriginatingAddress",
```

```

    "serverUrl": "localhost",
    "ackType": "Message"
  }
}

```

The SMS configuration is just an additional configuration property that can be used along native push configurations. You can also update the configuration by using the corresponding PUT and PATCH requests.

Admin API

HTTP Method	Action
<i>GET</i>	GET Push as API Service [page 499]
<i>POST</i>	POST Push as API Service [page 500]
<i>PUT</i>	PUT Push as API Service [page 501]
<i>PATCH</i>	PATCH Push as API Service [page 501]

[GET Push as API Service \[page 499\]](#)

Retrieves native push configuration based on application ID.

[POST Push as API Service \[page 500\]](#)

With a valid push configuration application registrations can be created as and when required using the POST method.

[PUT Push as API Service \[page 501\]](#)

Updates the native push configuration.

[PATCH Push as API Service \[page 501\]](#)

Updates an application's native push configuration. If the provided application does not exist, an application can be created and registered on the fly. This method merges a provided push configuration with any that already exist for an application.

[DELETE Push as API Service \[page 502\]](#)

Delete a native push configuration.

1.6.1.3.19.2.1 GET Push as API Service

Retrieves native push configuration based on application ID.

URI: `https://<mobile_services_host>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations`

HTTP Method *GET*

Request Parameters

Parameter	Description	Required	Parameter Type
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String

Response Status and Error Codes

Code	Description
200	Push configuration successfully retrieved
400	Invalid request.
403	Access denied for application.
404	Application not found.
500	Cannot retrieve configuration.

1.6.1.3.19.2.2 POST Push as API Service

With a valid push configuration application registrations can be created as and when required using the POST method.

URI: `https://<mobile services host>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
config	The push configuration.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String

Response Status and Error Codes

Code	Description
201	The application has been successfully configured for push.
304	A push configuration for this application already exists

Code	Description
400	Invalid push configuration.
403	Access denied for application.
500	Configuration not created.

1.6.1.3.19.2.3 PUT Push as API Service

Updates the native push configuration.

URI: `https://<mobile services host>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations`

HTTP Method *PUT*

Request Parameters

Parameter	Description	Required	Parameter Type
config	The push configuration.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String

Response Status and Error Codes

Code	Description
200	Push configuration successfully updated.
400	Invalid push configuration.
403	Access denied for application.
404	Application or push configuration not found.
500	Application configuration failed.

1.6.1.3.19.2.4 PATCH Push as API Service

Updates an application's native push configuration. If the provided application does not exist, an application can be created and registered on the fly. This method merges a provided push configuration with any that already exist for an application.

URI: `https://<mobile services host>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations`

HTTP Method *PATCH*

Request Parameters

Parameter	Description	Required	Parameter Type
config	The push configuration.	Yes	String
applicationId	The ID of the current application which should match the ID used for configuring your app for push notifications.	Yes	String

Response Status and Error Codes

Code	Description
200	Push configuration successfully updated.
400	Invalid push configuration.
403	Access denied for application.
404	Application or push configuration not found.
500	Configuration not created.

1.6.1.3.19.2.5 DELETE Push as API Service

Delete a native push configuration.

URI:https://<mobile services host>mobileservices/origin/hcpms/push/v1/admin/applications/{applicationId}/pushconfigurations

HTTP Method *DELETE*

Request Parameters

Parameter	Description	Required	Parameter Type
applicationId	The ID of the current application. This should match the application ID used while configuring push on HCPms for your app.	Yes	String

Response Status and Error Codes

Code	Description
204	Push configuration successfully deleted.

Code	Description
400	Invalid request.
403	Access denied for the application.
404	Application not found.
500	Push configuration deletion failed.

1.6.1.3.19.3 Back-end APIs

The notification user role is required to access back-end APIs.

The Push as API service provides services for the backend applications to trigger push notifications for either specific devices, or all devices of an application, or all devices of a certain user.

The Push as API service also allows you to send SMS notifications. You can send an SMS notification to the application instead of a native push notification. The Push as an API service allows you to use both SMS notifications as well as native push notifications.

Back-end API

HTTP Method	Action
<i>POST</i>	POST Push as API Service [page 503]
<i>GET</i>	GET Push as API Service [page 506]

[POST Push as API Service \[page 503\]](#)

Triggers a native push to users, an application, or a capability based on the request.

[GET Push as API Service \[page 506\]](#)

Fetches the status of a push notification.

1.6.1.3.19.3.1 POST Push as API Service

Triggers a native push to users, an application, or a capability based on the request.

Request (by User)

URI: `https://<mobile services host>/mobileservices/push/v1/backend/applications/{applicationId}/notifications/users`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
pushToUsersPayload	Target users	Yes	String
applicationId	Target application ID	Yes	String

Response Status and Error Codes

Code	Description
200	The push notification sent successfully
202	The push request has been successfully accepted.
304	The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.
400	The push request is syntactically incorrect.
500	Push failed.

Request (by Notifications)

You can trigger a native push to a specified device used by the given user. This back-end API can provide a device ID in addition to the username in case the push message should be received only on a certain device and not to all the devices linked to the given user.

URI: `https://<mobile services host>/mobileservices/push/v1/backend/applications/{applicationId}/users/{username}/devices/{deviceId}/notifications`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
pushPayload	The push notification payload.	Yes	String
applicationId	Target application ID.	Yes	String
username	The user to whom the device is assigned.	Yes	String
deviceId	The device ID of the device receiving the push notification.	Yes	String

Response Status and Error Codes

Code	Description
200	The push notification sent successfully
202	The push request has been successfully accepted.
304	The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.
400	The push request is syntactically incorrect.
500	Push failed.

Request (by Application)

URI: `https://<mobile services host>/mobileservices/push/v1/backend/applications/{applicationId}/notifications`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
pushToApplicationPayload	Notification payload.	Yes	String
applicationId	Target application ID.	Yes	String; Integer

Response Status and Error Codes

Code	Description
200	The push notification sent successfully
202	The push request has been successfully accepted.
304	The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.
400	The push request is syntactically incorrect.
500	Push failed.

Request (by Capability)

URI: `https://<mobile services host>/mobileservices/push/v1/backend/capabilities/{capabilityName}/notifications`

HTTP Method *POST*

Request Parameters

Parameter	Description	Required	Parameter Type
pushToCapabilitiesPayload	A list of capability users and a notification object.	Yes	String
capabilityName	Target capability.	Yes	String

Response Status and Error Codes

Code	Description
200	The push notification sent successfully
202	The push request has been successfully accepted.
304	The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.
400	The push request is syntactically incorrect.
500	Push failed.

1.6.1.3.19.3.2 GET Push as API Service

Fetches the status of a push notification.

URI: `https://<mobile services host>/mobileservices/push/v1/backend/notifications/{notificationId}/status`

HTTP Method [GET](#)

Request Parameters

Parameter	Description	Required	Parameter Type
notificationId	Push notification ID.	Yes	String

Response Status and Error Codes

Code	Description
200	Push notification status.
304	The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.
400	The request is missing its notification ID.
404	No push notification for the given ID found.
500	Push notification lookup failed.

1.6.1.3.20 Bundle Service

The Bundle Service API allows you to manage resource bundles directly for apps.

The bundle service supports two types of APIs:

- Runtime APIs
- Admin APIs

[Runtime APIs \[page 507\]](#)

The Bundle Service API allows you to manage resource bundles more directly for apps.

[Admin APIs \[page 508\]](#)

The Bundle Service API allows you to manage resource bundles more directly for apps.

Related Information

[Runtime APIs \[page 507\]](#)

[Admin APIs \[page 508\]](#)

1.6.1.3.20.1 Runtime APIs

The Bundle Service API allows you to manage resource bundles more directly for apps.

Runtime API

HTTP Method	Action
GET	GET Bundle Service API [page 507]

[GET Bundle Service API \[page 507\]](#)

Get the bundle metadata from the document repository of {applicationId}.

1.6.1.3.20.1.1 GET Bundle Service API

Get the bundle metadata from the document repository of {applicationId}.

HTTP Method: [GET](#)

Metadata Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata`

Response Status and Error Codes

Request Parameters

Parameter	Type	Description	Parameter Type
CmisRepository	Optional	A new entity	ApplicationId: string

Code	Description
200	The metadata of the document service
500	Internal server error

1.6.1.3.20.2 Admin APIs

The Bundle Service API allows you to manage resource bundles more directly for apps.

Admin API

HTTP Method	Action
GET	GET Bundle Service API [page 508]
POST	POST Bundle Service API [page 509]
DELETE	DELETE Bundle Service API [page 509]

[GET Bundle Service API \[page 508\]](#)

Gets the bundle metadata from the document repository of {applicationId}.

[POST Bundle Service API \[page 509\]](#)

This method allows you to create a Document Service repository.

[DELETE Bundle Service API \[page 509\]](#)

This method allows you to delete an entity in EntitySet CmisRepository.

1.6.1.3.20.2.1 GET Bundle Service API

Gets the bundle metadata from the document repository of {applicationId}.

HTTP Method: [GET](#)

Service Document

URL: https://<mobile_services_host>/mobileservices/origin/{origin}/bundles/v1/admin

Response Status and Error Codes

Code	Description
200	Service document

Code	Description
500	Unexpected error

Metadata Document

URL: `https://<mobile services host>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata`

Response Status and Error Codes

Code	Description
200	Metadata document
500	Unexpected error

1.6.1.3.20.2.2 POST Bundle Service API

This method allows you to create a Document Service repository.

HTTP Method: *POST*.

URL: `https://<mobile services host>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}`

Request Parameters

Parameter	Type	Description	Parameter Type
CmisRepository	Optional	A new entity	ApplicationId: string

Response Status and Error Codes

Code	Description
201	Created entity successfully
409	Document repository already exists
500	Unexpected error.

1.6.1.3.20.2.3 DELETE Bundle Service API

This method allows you to delete an entity in EntitySet CmisRepository.

HTTP Method: *DELETE*

URL: `https://<mobile_services_host>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId=' {ApplicationId} ',BundleName=' {BundleName} ',BundleVersion=' {BundleVersion} ')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	string

Response Status and Error Codes

Code	Description
202	Empty Response
404	Not Found
default	Unexpected error

1.6.1.3.21 Offline Store Upload Service

The Offline Store Upload API enables a client to upload offline store (database files) to the server. If Offline Store Upload is enabled for an application, you can upload the offline store (database files) into the server. The upload process creates a zip file that includes the uploaded database files, and is then saved into the server database.

Offline Store Upload supports two types of APIs:

- Runtime APIs
- Admin APIs

[Runtime APIs \[page 511\]](#)

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

[Admin APIs \[page 512\]](#)

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

Related Information

[Runtime APIs \[page 511\]](#)

[Admin APIs \[page 512\]](#)

1.6.1.3.21.1 Runtime APIs

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

Runtime APIs

HTTP Method	Action
POST	POST Offline Store Upload [page 511]

[POST Offline Store Upload \[page 511\]](#)

This method allows you to upload the offline store (database files) for an application. If Offline Store Upload is enabled for the specified application, then you can upload the offline store (database files) into the server. A zip file gets created to include the uploaded database files, and gets saved into the server database.

1.6.1.3.21.1.1 POST Offline Store Upload

This method allows you to upload the offline store (database files) for an application. If Offline Store Upload is enabled for the specified application, then you can upload the offline store (database files) into the server. A zip file gets created to include the uploaded database files, and gets saved into the server database.

HTTP Method: [POST](#).

URL: `https://<mobile services host>/mobileservices/application/<application>/offlinestoreupload/v1/runtime/application/<applicationId>/device/<deviceId>`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	The Application identifier.	String
deviceId	Required	The Device identifier. Its value gets saved with the uploaded log, if specified.	String

Parameter	Type	Description	Parameter Type
offlinestore	Required	<ul style="list-style-type: none"> The request body for the POST request. The content-type must be multipart/form-data. Multiple files can be included in one upload request. <p>Select form-data as Body Content Type.</p> <p>Provide Key name as file, and value refers to the database files.</p>	<p>OfflineStore</p> <p>An offline store consists of several databases, which needs to be uploaded in one upload request. Offline Store Upload API adds them into one zip file, and uploads the database files into the server. These uploaded database files are downloaded by administrator/developer together.</p>

Response Status and Error Codes

Code	Description
204	Offline Store uploaded.
400	Bad request. Such as request content is not multipart/form-data, or request does not include two files.
403	Offline Store upload is not enabled for the application.
404	No application found.
500	Unexpected error.

1.6.1.3.21.2 Admin APIs

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

Admin APIs

HTTP Method	Action
GET	GET Offline Store Upload [page 513]
POST	POST Offline Store Upload [page 515]
PATCH	PATCH Offline Store Upload [page 516]
DELETE	DELETE Offline Store Upload [page 516]

[GET Offline Store Upload \[page 513\]](#)

[POST Offline Store Upload \[page 515\]](#)

This method allows you to post a new entity to entity set PolicySet.

[PATCH Offline Store Upload \[page 516\]](#)

This method allows you to update the entity in EntitySet PolicySet.

[DELETE Offline Store Upload \[page 516\]](#)

1.6.1.3.21.2.1 GET Offline Store Upload

HTTP Method: *GET*

Service Document

URL: `https://<mobile_services_host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/`

Response Status and Error Codes

Code	Description
200	Service document.
500	Unexpected error

Metadata Document

URL: `https://<mobile_services_host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/$metadata`

Response Status and Error Codes

Code	Description
200	Metadata document.
500	Unexpected error

Entity set PolicySet

This method allows you to get the entities from the entity set PolicySet.

URL: `https://<mobile_services_host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/PolicySet`

Response Status and Error Codes

Code	Description
200	EntitySet PolicySet
404	Not found.
500	Unexpected error

PolicySet by Key

This method allows you to get the entity from PolicySet by key, and it returns the entity with the key from the PolicySet.

URL: `https://<mobile services host>/Admin/offlinestoreupload/PolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	String

Response Status and Error Codes

Code	Description
200	EntitySet PolicySet.
404	Not found.
500	Unexpected error.

Entity set OfflineStoreSet

This method allows you to get the entities from the entity set OfflineStoreSet.

URL: `https://<mobile services host>/Admin/offlinestoreupload/OfflineStoreSet`

Response Status and Error Codes

Code	Description
200	EntitySet PolicySet.
500	Unexpected error.

OfflineStoreSet by Key

This method allows you to get the entity from OfflineStoreSet by key, and returns the entity with the key from OfflineStoreSet

URL: `https://<mobile services host>/Admin/offlinestoreupload/OfflineStoreSet('{id}')`

Request Parameters

Parameter	Type	Description	Parameter Type
id	Required	key: id	String

Response Status and Error Codes

Code	Description
200	EntitySet OfflineStoreSet.
404	Not found.
500	Unexpected error.

OfflineStoreSet value

This method allows you to get the zip file, which includes the two databases for the offline store.

URL: `https://<mobile services host>/Admin/offlinestoreupload/OfflineStoreSet('{id}')/$value`

Request Parameters

Parameter	Type	Description	Parameter Type
id	Required	key: id	String

Response Status and Error Codes

Code	Description
200	The binary content of the zip file.
404	Not found.
500	Unexpected error.

1.6.1.3.21.2.2 POST Offline Store Upload

This method allows you to post a new entity to entity set PolicySet.

HTTP Method: [POST](#)

URL: `https://<mobile services host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/PolicySet`

Parameter	Type	Description	Parameter Type
Policy	Yes	The new entity. <ul style="list-style-type: none">It is the request body for POST request.If its content is in json format, then content-type of request must be application/json.If its content is xml/atom format, then the request's content-type must be application/xml.	<ul style="list-style-type: none">ApplicationId: stringLogUploadEnabled: boolean,nullExpiryTime: integer,nullMaxFileSize: integer,null

Response Status and Error Codes

Code	Description
201	Created entity

Code	Description
409	Offline Store Upload policy already exists
500	Unexpected error

1.6.1.3.21.2.3 PATCH Offline Store Upload

This method allows you to update the entity in EntitySet PolicySet.

HTTP Method: [PATCH](#)

URL: `https://<mobile services host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/PolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	String
Policy	Required	<p>The new entity.</p> <ul style="list-style-type: none"> It is the request body for PATCH request. If its content is in json format, then content-type of request must be application/json. If its content is xml/atom format, then the request's content-type must be application/xml. 	<ul style="list-style-type: none"> ApplicationId: string LogUploadEnabled: boolean,null ExpiryTime: integer,null MaxFileSize: integer,null

Response Status and Error Codes

Code	Description
204	Blank response
404	Not found
500	Unexpected error

1.6.1.3.21.2.4 DELETE Offline Store Upload

HTTP Method: [DELETE](#)

EntitySet PolicySet

This method allows you to delete the entity in EntitySet PolicySet.

URL: `https://<mobile services host>/mobileservices/application/{application}/offlinestoreupload/v1/admin/PolicySet('{applicationId}')`

Request Parameters

Parameter	Type	Description	Parameter Type
applicationId	Required	key: applicationId	string

Response Status and Error Codes

Code	Description
204	Blank response
404	Not found
500	Unexpected error

EntitySet OfflineStoreSet

This method allows you to delete the entity in EntitySet OfflineStoreSet.

URL: `https://<mobile services host>/Admin/offlinestoreupload/OfflineStoreSet('{id}')`

Request Parameters

Parameter	Type	Description	Parameter Type
id	Required	key: id	String

Response Status and Error Codes

Code	Description
204	Blank response
404	Not found
500	Unexpected error

1.6.1.3.22 Client Usage Report and User Feedback Upload Service

The Client Usage Report and User Feedback Upload service allows an application to upload client usage and user feedback charts for SAP Mobile Services.

Client Usage and User Feedback Upload Service supports Runtime APIs.

[Runtime APIs \[page 518\]](#)

The operations that are available in the Client Usage Report and User Feedback Upload Runtime APIs.

1.6.1.3.22.1 Runtime APIs

The operations that are available in the Client Usage Report and User Feedback Upload Runtime APIs.

Runtime API

HTTP Method	Action
POST	POST Client Usage Upload [page 518]
POST	POST User Feedback Upload [page 520]

[POST Client Usage Upload \[page 518\]](#)

Allows clients to upload client usage reports in JSON format.

[POST User Feedback Upload \[page 520\]](#)

Allows clients to upload user feedback in JSON format.

1.6.1.3.22.1.1 POST Client Usage Upload

Allows clients to upload client usage reports in JSON format.

❖ Example

The upload endpoint allows clients to upload client usage reports in the JSON format.

```
{
  "report": "reportUUID",
  (required)
  "appInfo": {
    "application":
"ClientUsageUpload0ae95adbc1e544388b46ab752688b86f",
    (required) " appVersion ":" < Version No. > version"
  },
  "deviceInfo": {
    "platform": "<Platform Name>platform",
    "platformVersion": "<Platform Version No.>platversion",
    "deviceModel": "<Device Model>Model",
    "deviceID ":" < Device ID > Device"(required)
  },
  "sessions": [{
    "sessionInfo": {
      "sessionId ":"
1e384510 ab8946848bc87fa266ccd53d"(required)
    },
    "events": [{
      "type ":" type1",
      "key ":" key1",
      "time": "2014-07-02T12:01:49.178+03",
      (required)
      "duration": "5.46760",
      "screen": "first screen",
```

```

        "view": "view1",
        "element": "element1",
        "action": "action1",
        "behavior": "login",
        "cases": "case1",
        "category": "category1",
        "result": "result1",
        "unit": "unit1",
        "measurement": "measurement1",
        "value ": "
        value1",
        "others ": "
        other param"
    }
  }
}

```

HTTP Method: *POST*

URL: `https://<mobile services host>/mobileservices/application/{applicationId}/clientusage/v1/runtime/clientusage/application/{applicationId}/device/{deviceId}`

Note

The device is optional (device/{deviceId}).

Parameter	Type	Description	Parameter Type
reports	Required	<ul style="list-style-type: none"> Refers to the uploaded client usage and user feedback charts file. Requires multipart/form-data request header. Supported in JAX-RS. 	file
applicationId	Required	<ul style="list-style-type: none"> ID of the current application. Must match the application ID that was used to configure the client usage and user feedback chart upload for your app. 	string

Response Status and Error Codes

Code	Description
201	The upload was successful.
400	The upload failed due to an incorrect payload.
403	The client usage and user feedback chart upload isn't enabled for the given application.
default	Unexpected error.

1.6.1.3.22.1.2 POST User Feedback Upload

Allows clients to upload user feedback in JSON format.

❖ Example

The upload endpoint allows clients to upload user feedback in JSON format.

```
{ "report": "reportUUID", "time": "2018-02-05T13:46:50.793+08",
  "anonymous": false, (required) "appInfo":
  { "application": "UserFeedbackUpload0ae95adb1e544388b46ab752688b86f",
    (required) "appVersion": "<Version No.>version" }, "deviceInfo":
  { "platform": "<Platform Name>platform", "platformVersion": "<Platform Version
    No.>platversion", "deviceModel": "<Device Model>Model", "deviceId": "<Device
    ID>Device"(required) }, "feedback": [ { "context": "app-defined context",
    "score": 10 (-99 <= score <= 99), "comment": "user-provided text" } ] }
```

HTTP Method: *POST*

URL: *<mobile services host>/mobileservices/application/{applicationId}/clientfeedback/v1/runtime/clientfeedback/application/{applicationId}*

Parameter	Type	Description	Parameter Type
applicationId	Required	<ul style="list-style-type: none">ID of the current application.Must match the application ID that was used to configure the user feedback upload for your app.	string

Response Status and Error Codes

Code	Description
201	The upload was successful.
400	The upload failed due to an incorrect payload.
default	Unexpected error.

1.6.1.3.23 Role Service

This service allows you to get the logical roles that are assigned to the current user, which you can use to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

Prerequisite: Access Control Policy

In the access control policy, define which roles are relevant for the current mobile application. The Role service checks each role in the Access Control policy, and returns those that are assigned to the current user

Note

If the user does not have the required role and tries to register for an application, the access control policy returns a 403 error message.

The Role service supports Runtime APIs.

[Runtime APIs \[page 521\]](#)

The Role service API allows the application to retrieve user and roles for the currently authorized user, using the SCIM protocol.

Related Information

[Runtime APIs \[page 521\]](#)

1.6.1.3.23.1 Runtime APIs

The Role service API allows the application to retrieve user and roles for the currently authorized user, using the SCIM protocol.

Runtime API

HTTP Method	Action
GET	GET Role Service [page 521]

[GET Role Service \[page 521\]](#)

Gets all the logical roles belonging to the current user.

1.6.1.3.23.1.1 GET Role Service

Gets all the logical roles belonging to the current user.

HTTP Method: [GET](#)

URL: `https://<mobile services host>/mobileservices/application/{appId}/roleservice/application/{appId}/v2/Me`

Request Parameters

Parameter	Type	Description	Parameter Type
appId	Required	The current application to use the role service.	string

Response Parameters

Parameter	Type	Description	Parameter Type
id	Mandatory	The user's ID if included in SAML or OAuth authentication.	string
userName	Mandatory	The user's user name if included in SAML or OAuth authentication.	string
roles	Optional	The user's role if included in SAML or OAuth authentication.	string
familyName	Optional	The user's family name if the "family_name" attribute is included in SAML or OAuth authentication.	string
givenName	Optional	The user's given name if the "given_name" attribute is included in SAML or OAuth authentication.	string
emails	Optional	The user's email address if the "email" attribute is included in SAML or OAuth authentication.	string

Response Status and Error Codes

Code	Description
200	Retrieval of the roles associated with the user is successful.
403	The user does not have the required authorization to access this application.
404	Application ID does not exist.
412	The application is not enabled for access control.

1.6.1.4 Application Connection Properties

Describes application connection properties, and indicates whether the properties are read-only or nullable from the HTTP client.

Note

If you attempt to modify a read-only property, the client application throws the following exception: HTTP 403 - The property "XXX" cannot be updated by a client application.

Onboarding Version 1 or Later

Application Connection Properties: Uncategorized

Property Name	Description	Type	Read-only?	Is Nullable?
ETag	Specifies the version identifier.	String	Yes	No
ApplicationConnectionId	ID that uniquely identifies an application. Usually generated by the server and in the format of a GUID.	String	Yes	No

Application Connection Properties: Android Push

Property Name	Description	Type	Read-only?	Is Nullable?
AndroidGcmPushEnabled	Indicates if Google Cloud Messaging (GCM) push notifications are enabled and configured for this application.	Boolean	No	No
AndroidGcmRegistrationId	The registration ID that the device acquires from Google during GCM registration.	String	No	Yes
AndroidGcmSenderId	GCM sender ID used by SAP Mobile Server to send notifications. Used by the client to register for GCM.	String	Yes	Yes

Application Connection Properties: Apple Push

Property Name	Description	Type	Read-only?	Is Nullable?
ApnsPushEnable	Indicates if push notification using APNs is enabled or not.	Boolean	No	No
ApnsDeviceToken	The Apple push notification service token.	String	No	Yes

Application Connection Properties: Application Settings

Property Name	Description	Type	Read-only?	Is Nullable?
CustomizationBundleId	The application configuration (customization resource bundles) associated with the application.	String	Yes	Yes
ApplicationVersion	The version number of the registered application.	String	No	Yes

Property Name	Description	Type	Read-only?	Is Nullable?
ClientSdkVersion	The version number of the SDK for the registered application.	String	No	Yes

Application Connection Properties: BlackBerry Push

Property Name	Description	Type	Read-only?	Is Nullable?
BlackberryPushEnabled	Indicates if BlackBerry push notifications are enabled and configured for this application.	Boolean	No	No
BlackberryDevicePin	Every Blackberry device has a unique permanent PIN. During initial connection and settings exchange, the device sends this information to the server.	String	No	Yes
BlackberryBESListenerPort	The listener port for BES notifications.	Int32	No	No

Application Connection Properties: Windows Push

Property Name	Description	Type	Read-only?	Is Nullable?
WnsChannelURI	The WNS Channel URI as provided by WNS during push registration.	String	No	Yes
WnsPushEnable	Indicates if push notification using Wns is enabled or not.	Boolean	No	No

Application Connection Properties: MPNS Push

Property Name	Description	Type	Read-only?	Is Nullable?
MpnsChannelURI	The MPNS Channel URI as provided by MPNS during push registration.	String	No	Yes
MpnsPushEnable	Indicates if push notification using Mpns is enabled or not.	Boolean	No	No

Application Connection Properties: Capabilities

Property Name	Description	Type	Read-only?	Is Nullable?
CapabilitiesPasswordPolicy	Updates the password policy for an application connection.	Boolean	No	No

Application Connection Properties: Custom Settings

Property Name	Description	Type	Read-only?	Is Nullable?
CustomCustom1	Available for free use.	String	No	Yes
CustomCustom2	Available for free use.	String	No	Yes
CustomCustom3	Available for free use.	String	No	Yes
CustomCustom4	Available for free use.	String	No	Yes

Application Connection Properties: Device Information

Property Name	Description	Type	Read-only?	Is Nullable?
DeviceModel	The manufacturer of the registered mobile device.	String	No	Yes
DeviceType	The type of device. Supported device types are: WinMobile, WinSmartPhone, Windows, iPhone, iPad, iPod, iOS, BlackBerry, Android, Rim6, Windows8, WinPhone8, Windows81, WinPhone81.	String	No	Yes
DeviceSubType	The device subtype of the device.	String	No	Yes
DevicePhoneNumber	The phone number associated with the registered mobile device. This phone number is used when sending SMS text messages to a device. The phone number must be provided in international format, e.g. "+491712345678" or "+15551231234".	String	No	Yes
DeviceIMSI	The International Mobile Subscriber identity, which is a unique number associated with all Global System for Mobile communication (GSM) and Universal Mobile Telecommunications System (UMTS) network mobile phone users.	String	No	Yes

Application Connection Properties: Password Policy

Property Name	Description	Type	Read-only?	Is Nullable?
PasswordPolicyEnabled	Specifies whether password policies are enabled.	Boolean	Yes	No
PasswordPolicyDefaultPasswordAllowed	Specifies whether default passwords are allowed.	Boolean	Yes	No
PasswordPolicyMinLength	Specifies how long the password chosen by the user must be.	Int32	Yes	No
PasswordPolicyDigitRequired	Specifies whether the password must contain digit(s).	Boolean	Yes	No
PasswordPolicyUpperRequired	Specifies whether the password must contain uppercase characters.	Boolean	Yes	No
PasswordPolicyLowerRequired	Specifies whether the password must contain lowercase characters.	Boolean	Yes	No
PasswordPolicySpecialRequired	Specifies whether the password must contain non-alphanumeric characters.	Boolean	Yes	No
PasswordPolicyExpiresInNDays	Specifies the number of days the existing password can be used before it must be changed by the user.	Int32	Yes	No
PasswordPolicyMinUniqueChars	Determines how many unique characters must be used in the password.	Int32	Yes	No
PasswordPolicyLockTimeout	Determines how long a successfully unlocked data vault will remain open. When the timeout expires, the vault is locked, and the user must re-enter the vault password to resume using the application.	Int32	Yes	No

Property Name	Description	Type	Read-only?	Is Nullable?
PasswordPolicyRetry-Limit	Determines how long a successfully unlocked data vault will remain open. When the time-out expires, the vault is locked, and the user must re-enter the vault password to resume using the application.	Int32	Yes	No

Application Connection Properties: Proxy

Property Name	Description	Type	Read-only?	Is Nullable?
ProxyApplicationEndpoint	The URL pointing to the EIS.	String	Yes	Yes
ProxyPushEndpoint	The SAP Mobile Platform URL for sending out notifications.	String	Yes	Yes

Application Connection Properties: Usage

Property Name	Description	Type	Read-only?	Is Nullable?
MaxConnectionWaitTimeForClientUsage	Determines how long a connection exists for client usage.	Int32	Yes	Yes
EnableAppSpecific-ClientUsageKeys	Determines if the application developer can use custom information in Usage Data Collection.	Boolean	Yes	Yes

Application Connection Properties: Log

Property Name	Description	Type	Read-only?	Is Nullable?
UploadLogs	Specifies whether log upload is enabled for this app and device registration.	Boolean	Yes	Yes
LogEntryExpiry	Specifies the maximum time the logs would be kept on the device before they get removed.	Int32	Yes	Yes

Application Connection Properties

Property Name	Description	Type	Read-only?	Is Nullable?
E2ETraceLevel	The log level to be used for End 2 End traces.	String	Yes	Yes

Property Name	Description	Type	Read-only?	Is Nullable?
PublishedToMobile-Place	Determines if the configuration is currently shared through mobile place or not.	Boolean	Yes	Yes
FeatureVectorPolicyAIIEnabled	Determines if the hybrid capability policy is enabled or not. If enabled then the administrator can select which features to allow to work per application. If disabled all features will work.	Boolean	Yes	Yes

Onboarding Version 3 or Later

Application Connection Properties: Capability

Property Name	Description	Type	Read-only?	Is Nullable?
Category	The capability category.	String	Yes	No
CapabilityName	Name of the capability.	String	Yes	No
ApplicationConnectionId	ID that uniquely identifies an application.	String	Yes	No
CapabilityValue	Value of the capability.	String	Yes	Yes

Application Connection Property: Form

Property Name	Description	Type	Read-only?	Is Nullable?
FormFactor	The form factor of the device. Mostly used in combination with Capability based push.	String	Yes	No

Onboarding Version 4 or Later

Application Connection Property: UserName

Property Name	Description	Type	Read-only?	Is Nullable?
UserName	Specifies the user name that was used during registration	String	Yes	No

Application Connection Property: Application Settings

Property Name	Description	Type	Read-only?	Is Nullable?
UserLocale	Specifies the users preferred language.	String	No	Yes
TimeZone	Specifies the time zone.	String	No	Yes
CreatedAt	Indicates the time the application was created.	DateTime	Yes	No
PushGroup	Custom field to specify a target audience for push information.	String	No	Yes
Email	Users email address.	String	No	Yes

1.7 Migration

With minimal disruption, you can migrate applications from the SAP Mobile Services,Neo environment to the Cloud Foundry environment. You can also manually migrate applications from SAP Mobile Platform to SAP Mobile Services.

- **Migration Scenario 1 – SAP Mobile Services Neo to Cloud Foundry**
Use this scenario if you are currently using SAP Mobile Services on a Neo landscape, and plan to migrate to a Cloud Foundry landscape. See *Migrate: SAP Mobile Services Neo to Cloud Foundry*.
- **Migration Scenario 2 – SAP Mobile Platform to SAP Mobile Services**
Use this scenario if you are currently using SAP Mobile Platform, and plan to migrate to the SAP Mobile Services environment. See *Migrate: SAP Mobile Platform to SAP Mobile Services*.

Note

If you are already using SAP Mobile Services Cloud Foundry, and have not yet upgraded your apps from a mobile services model to the Mobile Services Cloud Foundry service model, see *Upgrading Apps to Mobile Services Cloud Foundry Service*.

[Migrate: SAP Mobile Services Neo to Cloud Foundry \[page 530\]](#)

Provides information about migrating from SAP Mobile Services, Neo environment to Cloud Foundry.

[Migrate: SAP Mobile Platform to SAP Mobile Services \[page 533\]](#)

Provides information about migrating from SAP Mobile Platform to SAP Mobile Services.

[Upgrading Apps to Mobile Services Cloud Foundry Service \[page 535\]](#)

Upgrade your apps from a mobile services model to the Mobile Services Cloud Foundry service model. This makes it easier to manage customer apps in the Cloud Foundry provider space.

[New Feature Comparison: Cloud Foundry and Neo \[page 536\]](#)

Describes new SAP Mobile Services features that have been added to Cloud Foundry, and not Neo.

1.7.1 Migrate: SAP Mobile Services Neo to Cloud Foundry

Provides information about migrating from SAP Mobile Services, Neo environment to Cloud Foundry.

SAP Mobile Services on Cloud Foundry is largely identical to its Neo counterpart. This allows you to migrate mobile applications from Neo to Cloud Foundry. Some features are not supported for Mobile Services on Cloud Foundry. Keep in mind the following differences with Cloud Foundry.

Missing Functionality in SAP Mobile Services Cloud Foundry

- Mobile Application security configuration does not support "No Authentication". All interaction from a mobile application must be made in an authenticated manner.
As an alternative, Mobile Services on Cloud Foundry provides the security configuration "API Key Only" which uses a common API Key per application for all users connecting to Mobile Services and does not require user-specific authentication.
- The destination SSO Mechanism 'SAPAssertionSSO' type is not supported in Cloud Foundry.
- Mobile Application security configuration does not support certificate based authentication. SAP Mobile Services Cloud Foundry does not support establishing an SSL connection using client certificates (mutual authentication). You can still leverage certificates deployed to mobile applications with your identity provider of choice. Please use authentication methods OAuth (preferred) or SAML in the security configuration. The user authentication occurs in a Web View, which can present the certificate to the identity provider if requested. This avoids the need for any user input, while still leveraging an existing certificate infrastructure.
- One-time passcodes in combination with Basic Authentication is not supported. Instead you can configure one-time passcodes with two-factor authentication right in SAP Identity Authentication Service. This is supported for the Mobile Services authentication methods SAML and OAuth.
- Secure login server is not supported.
- Document repository is not supported.
As an alternative, you can create a destination to a back-end system using existing Cloud Foundry service instances. See [Creating a Destination with Existing Service Instances](#), and search for "Document service instances" to learn more.

Mobile Services Cockpit Changes

If you are very familiar with the SAP Mobile Services Neo cockpit, you will find that there are several differences in the SAP Mobile Services cockpit. The major changes are:

- Access to Mobile Services cockpit is authenticated with the SAP ID Service (Default identity provider). The Trust Configuration from the Cloud sub-account is only used at runtime (when a user connects through a mobile application).
- Destinations are no longer configured at the global level, but at the application level.
- Network traces are configured and accessed as a feature at the application level.
- The list of onboarded users and devices is part of the Mobile Settings Exchange feature (mandatory feature for each mobile application). You can see the list of registered users and devices in this feature under [User Registrations](#).

- There is no more global log configuration page. Instead you can enable detailed event logs for each mobile application and feature separately. When enabled, DEBUG and INFO event log messages are logged and available under ► [Analytics](#) ► [Logs](#) (just as they are on Neo).
- The Security configuration of a mobile application is no longer handled as a feature, but as a separate section (tab) for the mobile application.
- Test push messages are now triggered from within the Mobile Push Notification feature of a mobile application.
- To enable basic authentication you need to establish trust between Mobile Services and the sub-account. This must be done once for each Space in which Mobile Services is used, and configured for supporting Basic authentication.
- Uploaded client logs (mobile application logs) are now accessed from with the Mobile Client Log Upload feature of a mobile application.

Migration of Mobile Application Configurations

Once you become familiar with some of the changes in Mobile Services cockpit operations, replicating mobile application configurations between Neo and Cloud Foundry should be straightforward. Ideally, replicate a few applications and accounts manually to become familiar with the Mobile Services cockpit.

You can recreate the application manually, or export the application configuration on Neo and import the configuration on Cloud Foundry. The latter process is described in *Importing Application Configurations*. Please be aware of the caveats described on that page. Typically some manual adjustments are required after you import the application (rarely are adjustments not needed).

User (App) Migration

Mobile applications developed for SAP Mobile Services Neo are generally compatible with the Cloud Foundry counterpart with the aforementioned exceptions.

The application status must be reset when transitioning from Neo to Cloud Foundry (similar to transitioning from one Neo landscape to another).

This should happen automatically when resetting the application and reconfiguration with a new URL. Users need to ensure that there is no unsynchronized data left in the local Offline databases before the application is reset.

The mobile application URL of Neo and Cloud Foundry is different. In Cloud Foundry, each mobile application has its own, unique hostname to which to connect, as shown on the APIs page of an application. Also, the OAuth specific URLs of Neo and Cloud Foundry are different (authorization and token endpoints).

- In case these URLs are hard-coded in the application binary that is distributed to your users (for example, via public app stores), then you will need to roll out a new version of the application. If this is distributed through a mobile device management tool, it also allows the phased migration of users between Neo and Cloud Foundry, and ability to observe the system before transitioning all users.
- If the application is configured by scanning a QR code, the QR code reflecting the new configuration can be downloaded from the Mobile Services cockpit and scanned by users after having the application reset.

- Applications that use Discovery services to retrieve the onboarding details reload the configuration after the application is reset. New users automatically read the new configuration. This requires you to register the email domain with the new Mobile Services instance on Cloud Foundry and publish the configuration.

Migration Check-list

- Ensure (with information on this page) that Mobile Services Cloud Foundry supports all your requirements.
- Enable [Mobile Services](#) via Entitlements on your Global Account and assign it to the desired Cloud Foundry sub-account. Refer to the *Get Started* section for further details.
- Create at least one space in the designated sub-account.
- Optional: connect SAP cloud connector to the Cloud Foundry sub-account (the same cloud connector can be connected to several SAP BTP sub-accounts).
- Replicate mobile application configuration (see this page for further details).
- Enable "Detailed Event Logs" to capture any discrepancies early on.
- Test the mobile application by changing the URL to point to the new landscape.
- Verify the Logs in Mobile Services for any potential problems.
- Follow any QA procedures that are specific to your organization.
- Migrate a few users manually and make sure the application and back ends are behaving correctly before rolling out a large user base.
- Plan the roll-out of the application. See further details on this page for the various options. Users must be aware that unsynchronized work (Offline) cannot be carried forward from Neo to Cloud Foundry. The application should be in a synchronized status before reconfiguration.

Note

You can enable Mobile Services in any number of Cloud Foundry Spaces in a sub-account. Those instances of Mobile Services are completely isolated from each other.

This can be an appropriate approach to isolate various stages of your development process (dev/test/QA/prod). Keep in mind, however, that certain elements are shared on a sub-account level. This includes connected Cloud Connectors, Identity Providers, and general trust configuration.

In most cases the better approach is to also isolate stages at a sub-account level, as you are familiar with on Neo.

Related Information

[Get Started \[page 18\]](#)

[Importing Application Configurations \[page 228\]](#)

1.7.2 Migrate: SAP Mobile Platform to SAP Mobile Services

Provides information about migrating from SAP Mobile Platform to SAP Mobile Services.

A completely seamless migration is impossible, due to the environmental differences between SAP Mobile Platform and SAP Mobile Services. A migration affects both the local architecture and the user experience. The following table describes the source and corresponding target landscape for an SAP Mobile Platform application.

Pre-Migration Landscape	Post-Migration Landscape (SAP Mobile Services)
	SAP cloud connector
SAP Mobile Platform 3.x installed	SAP Mobile Services installed
SAP NetWeaver Gateway provides OData services for the mobile application to be migrated (on-premise)	SAP NetWeaver Gateway provides OData services for the mobile applications to be migrated (on-premise)
Mobile applications are hybrid Android, iOS, or Windows 8.1 applications, which use the Mobile Application Framework (MAF) Logon plug-in.	Mobile applications are hybrid Android, iOS, or Windows 8.1 applications, which use the Mobile Application Framework (MAF) Logon plug-in.
Mobile user authentication uses basic HTTP against the SAP NetWeaver Gateway system on-premise and external OData services.	Mobile user authentication uses basic HTTP against the SAP NetWeaver Gateway system on-premise and external OData services.

Applications You Cannot Migrate

Migration is not possible for:

- Agentry-based applications
- Mobiliser-based applications
- Mobile applications that require custom OSGi bundles; in this scenario, you must migrate the code-base bundle to other tools.
- Applications that are based on mobile business object (MBO) technology
- Applications that use custom OSGi authentication modules
- Short Message Service (SMS) based applications

[Migrating to SAP Mobile Services \[page 534\]](#)

Manually migrate an SAP Mobile Platform application to the Mobile Services.

1.7.2.1 Migrating to SAP Mobile Services

Manually migrate an SAP Mobile Platform application to the Mobile Services.

Procedure

1. Collect the required information for migrating from the source system:
 - Back-end service URL
 - Back-end authentication service URL
 - Application ID
 - URL rewrite options
 - Proxy type
 - Authentication option
 - Client password policy
 - Client log policy
 - Push settings
 - Client resources
 - Application-specific settings
 2. Install SAP Cloud Connector. See [Installation](#) See .
 3. On the [Access Control](#) in SAP cloud connector, allow list the necessary back-end service URLs.
 - To use an internal OData URL, select [OnPremise](#) as the proxy type.
 - To use an Internet OData Service URL, select [Internet](#) as the proxy type.

See [Configure Access Control \(HTTP\)](#).
 4. In SAP Mobile Services, create an application. See [Configuring Applications](#).
 5. (Optional if you use your mobile app for testing) Test the application configuration with REST client. See [REST API Application Development Overview](#).
- Ensure that the initial server configuration is complete and working.
6. Migrate the client application to the latest SAP Mobile SDK. See SAP BTP SDK in [SAP Mobile Platform SDK](#).
 7. Check the network connectivity settings of the device.
 8. Use corporate WiFi to connect to SAP Mobile Platform.
 9. Configure and enable WiFi with Internet access to connect to SAP Mobile Services.
 10. Before onboarding the client application on the device, verify that Mobile Services cockpit is reachable.
 11. For applications with offline features, check the proxy setting in the client application source code.

Note

For example, if the offline store cannot open or has a network error during runtime for ODataOfflineStoreOption object in the following format:
`MyODataOfflineStoreOptions.extraStreamParms="proxy_host=myproxy;proxy_port=8080";`

12. Run a complete regression test.

Note

Already enrolled apps will not automatically be registered with SAP Mobile Services; you must reregister these apps with the new URL

Related Information

[Configuring Applications \[page 45\]](#)

1.7.3 Upgrading Apps to Mobile Services Cloud Foundry Service

Upgrade your apps from a mobile services model to the Mobile Services Cloud Foundry service model. This makes it easier to manage customer apps in the Cloud Foundry provider space.

Mobile Services is now available in a new model: unlike before when 14 individual Cloud Foundry services needed to be enabled to use the full set of features, all of them are now available under a single service called "Mobile Services". In addition the required resources in your account have been reduced.

Mobile Services is a standard Cloud Foundry service that allows users to consume mobile services through a mobile-approuter deployed under a provider space instead of a customer space. Feature services, such as storage, proxy and so forth, are managed as internal services instead of Cloud Foundry services.

When you define a new mobile application in Mobile Services cockpit a Mobile Services service instance is created automatically. For existing mobile applications, you must take action to upgrade the existing mobile applications to the new model. The process is automated. For a time you can operate applications without upgrading, but in the future the old method will not be supported.

Step 1: Enable Mobile Services Cloud Foundry Service

When you access Mobile Services cockpit and Mobile Services detects that the new Mobile Services Cloud Foundry service is not enabled in the customer space, you see the banner *Mobile Application Service is not enabled. The service is required for upgrade. Check SAP Help Portal for instructions.*

Select [SAP Help Portal](#) for instructions on how to enable the service in SAP BTP.

Step 2: Upgrade Existing Apps to Mobile Services Cloud Foundry Service

Once the Mobile Services service is enabled in the space and you access Mobile Services cockpit, you see the banner *Application upgrade is required. Click Here to see details.*

The [Upgrade](#) pop-up window provides more details. Select [Upgrade Now](#). During the upgrade, no other actions can be carried out in the cockpit. Alternatively you can select [Upgrade Later](#) to delay the upgrade, but in the future the upgrade must be done.

During the upgrade process, a progress bar keeps you informed of the status. Processing time depends on the number of apps that need to be upgraded, but it is typically fairly fast (minutes versus hours).

If the upgrade fails, the [Warning](#) dialog informs you and offers several choices. You can:

- Select [Retry](#) to resume the upgrade process at the point that it previously failed.
- Select [Go to Event Logs](#) to navigate to the logs page and obtain more information.
- You can also select [Cancel](#) to close the dialog, and take care of the problem later. Users can still operate the applications that have not been upgraded, as well as those that have been upgraded, but you cannot view or manage migrated applications from Mobile Services cockpit until all applications have been upgraded successfully. Some features may not work correctly from Mobile Services cockpit (such as SAP Mobile Cards or cloud build and security service settings), if the standalone service instance has not been upgraded. Once all existing applications have been upgraded, you can again view and manage the upgraded apps in the cockpit.

Post-Upgrade

Once all applications have been upgraded successfully, these properties no longer appear when you define a new application:

- [Client Connection Timeout](#)
- [Session Timeout](#)
- [Enable Compression](#)
- [Zero Downtime Maintenance](#)

These properties remain:

- [Backend Connection Timeout](#)
- [Minimum Compression Size](#)

Otherwise there are no other Mobile Services cockpit changes related to this upgrade.

1.7.4 New Feature Comparison: Cloud Foundry and Neo

Describes new SAP Mobile Services features that have been added to Cloud Foundry, and not Neo.

Mobile Services (Cloud Foundry) is the mainstream version, and most new features of Mobile Services will only be available on Cloud Foundry.

Following is a list of features available on Cloud Foundry but not on Neo. The list contains only major runtime features that directly impact mobile application functioning. Most user interface enhancements to Mobile Services cockpit, including the analytics dashboard and others, are not covered.

1. PKCE (Proof Key for Code Exchange) support for OAuth grant flow
2. Localized push notification messages

3. Micro App WeChat support (regular / social; China only)
4. Digitally signed QR code support for onboarding
5. Offline OData V4
6. Crash log analysis
7. Client Usage Feedback Service
8. Augmented Reality
9. Cross Context SSO support
10. App Catalog
11. Micro App DingTalk support (not available in all countries/regions)
12. Service Keys support
13. W3C push support
14. Customized Mobile Development Kit Client support in Cloud Build Service
15. Theming and Branding Support
16. Push to topics
17. iOS and Android Attestation Support

1.8 Glossary

Defines terms and components for SAP Mobile Services.

anonymous user	A user who can access the system without providing identification.
Apple Push Notification Service (APNs)	A free service provided by Apple for devices running iOS. The APNs pushes notifications from a provider to a device, which means applications need not operate as active listeners for those notifications.
application user	A distinct set of identities (identified or anonymous) that are in contact with the system by using an application. In Mobile Services Cockpit, an application user is the distinct list of names under which a user has been identified to the system. An application user may also be a user (identified or anonymous) that has been associated with an application ID.
back end	A system that provides a data source, such as a database or Web service.
certificate	A digital security mechanism attached to an electronic message that verifies the identity of a specific user.
client application	In SAP Mobile Services, the software that runs on a smart phone, tablet computer, or other mobile device. See mobile application.
client resources	Also known as resource bundles. Containers used by applications to download dynamic configurations, styles, or content from the cloud.
connection	Configuration details and credentials that are required to connect to a database, Web service, or other back end.
data vault	Provides encrypted storage of occasionally used, small pieces of data from multiple operational systems.

deploy	To upload a computer program or development unit from a development state to a server, moving it from a packaged or assembled form to an operational working state that can be consumed. SAP Mobile Services can then make the unit accessible to users via a client application that is installed on a mobile device.
device application	A software application that runs on a mobile device. See mobile application.
discovery service	Provides the configuration information necessary for a user to enroll a device with SAP Mobile Secure. Allows you to distribute initial configuration data to mobile apps to enhance the user onboarding process.
export	The movement of mobile objects from a system so they can be imported into another system. Typically performed by the SAP BTP mobile services administrator.
Firebase Cloud Messaging (FCM)	A free service offered by Google for sending messages to Android devices. Requires an API key to allow SAP Mobile Services to send push notifications over FCM.
hybrid application	An application developed using Web technologies, such as HTML5 and JavaScript, that runs within a native application on the device. The container provides the Web application with access to native device capabilities through an exposed JavaScript API.
keystore	The location in which encryption keys, digital certificates, and other credentials in either encrypted or unencrypted keystore file types are stored for SAP BTP mobile services runtime components. See truststore.
Lightweight Directory Access Protocol (LDAP)	An application protocol for accessing, querying, and modifying data in distributed directory services.
mobile application (mobile app)	A software application designed to run on smart phones, tablet computers, and other mobile devices.
Mobile Services Cockpit	A Web-based interface in SAP Mobile Services for creating and administering mobile applications, registering users, creating and maintaining connections, and performing administration tasks related to reporting, logging, and onboarding.
monitoring	A SAP Mobile Services feature that allows administrators to identify areas of weakness or periods of high activity in a particular area, as well as overall system health. Use for system diagnostics or for troubleshooting.
Microsoft Push Notification Service for Windows Phone (MPNS)	A free service that enables you to send push notification messages to Windows Phone 7+ and Windows Phone 8.0 apps.
Open Data Protocol (OData)	Provides standard create, read, update, and delete (CRUD) access to a data source via a web site. OData is similar to JDBC and ODBC, although not limited to SQL databases.
OData proxy	A connection to the mobile server that funnels OData service requests through the platform, giving administrators and developers more control by forcing only allow-listed endpoints to be accessible from the application. Also restricts who is able to access the endpoint, based on security mechanisms that are built into the platform.



onboarding	The enterprise-level activation of an authentic device, a user, and an application entity as a combination in SAP Mobile Services.
Security Assertion Markup Language(SAML)	An XML-based open standard data format for exchanging authentication and authorization data between an identity provider and a service provider.
SAP Fiori	The user experience (UX) for SAP software; represents a personalized, responsive, and simple user experience across devices and deployment options.
SAP BTP	Platform as a Service (PaaS) offering from SAP; enables customers and developers to build, extend, and run applications on SAP in the cloud.
SAP Mobile Place	An SAP mobile application management offering that is a brandable, localizable, and secure enterprise app store, making it easy for companies to push their mobile apps to employees, business partners, and consumers.
SAP Mobile Secure	A cloud-based SAP Enterprise Mobility Management (EMM) offering.
schedule	The definition of a task (such as the collection of a set of statistics) and the time interval during which the task must execute using SAP BTP mobile services.
System for Cross-domain Identity Management (SCIM)	An open standard that connects SAP Mobile Services to a back-end authentication user store.
security configuration	The mechanism within SAP BTP that enforces application authentication and authorization. The security configuration points the platform to an underlying user store (a repository, such as Active Directory or an LDAP server) to perform authentication and authorization services.
single sign-on (SSO)	A credential-based authentication mechanism for accessing multiple, but independent, software systems using a single logon.
truststore	The location in which certificate authority (CA) signing certificates are stored. See keystore.
Windows Push Notification Service (WNS)	A free service that enables third-party developers to send toast, tile, badge, and raw updates from their own cloud service to Windows Store apps. All modern UI apps can receive notifications via WNS, but not traditional desktop applications. See MPNS for information about push notification service to Windows Phone.

Important Disclaimers and Legal Information

Hyperlinks

Some links are classified by an icon and/or a mouseover text. These links provide additional information.

About the icons:

- Links with the icon  : You are entering a Web site that is not hosted by SAP. By using such links, you agree (unless expressly stated otherwise in your agreements with SAP) to this:
 - The content of the linked-to site is not SAP documentation. You may not infer any product claims against SAP based on this information.
 - SAP does not agree or disagree with the content on the linked-to site, nor does SAP warrant the availability and correctness. SAP shall not be liable for any damages caused by the use of such content unless damages have been caused by SAP's gross negligence or willful misconduct.
- Links with the icon  : You are leaving the documentation for that particular SAP product or service and are entering an SAP-hosted Web site. By using such links, you agree that (unless expressly stated otherwise in your agreements with SAP) you may not infer any product claims against SAP based on this information.

Videos Hosted on External Platforms

Some videos may point to third-party video hosting platforms. SAP cannot guarantee the future availability of videos stored on these platforms. Furthermore, any advertisements or other content hosted on these platforms (for example, suggested videos or by navigating to other videos hosted on the same site), are not within the control or responsibility of SAP.

Beta and Other Experimental Features

Experimental features are not part of the officially delivered scope that SAP guarantees for future releases. This means that experimental features may be changed by SAP at any time for any reason without notice. Experimental features are not for productive use. You may not demonstrate, test, examine, evaluate or otherwise use the experimental features in a live operating environment or with data that has not been sufficiently backed up.

The purpose of experimental features is to get feedback early on, allowing customers and partners to influence the future product accordingly. By providing your feedback (e.g. in the SAP Community), you accept that intellectual property rights of the contributions or derivative works shall remain the exclusive property of SAP.

Example Code

Any software coding and/or code snippets are examples. They are not for productive use. The example code is only intended to better explain and visualize the syntax and phrasing rules. SAP does not warrant the correctness and completeness of the example code. SAP shall not be liable for errors or damages caused by the use of example code unless damages have been caused by SAP's gross negligence or willful misconduct.

Bias-Free Language

SAP supports a culture of diversity and inclusion. Whenever possible, we use unbiased language in our documentation to refer to people of all cultures, ethnicities, genders, and abilities.

© 2024 SAP SE or an SAP affiliate company. All rights reserved.

No part of this publication may be reproduced or transmitted in any form or for any purpose without the express permission of SAP SE or an SAP affiliate company. The information contained herein may be changed without prior notice.

Some software products marketed by SAP SE and its distributors contain proprietary software components of other software vendors. National product specifications may vary.

These materials are provided by SAP SE or an SAP affiliate company for informational purposes only, without representation or warranty of any kind, and SAP or its affiliated companies shall not be liable for errors or omissions with respect to the materials. The only warranties for SAP or SAP affiliate company products and services are those that are set forth in the express warranty statements accompanying such products and services, if any. Nothing herein should be construed as constituting an additional warranty.

SAP and other SAP products and services mentioned herein as well as their respective logos are trademarks or registered trademarks of SAP SE (or an SAP affiliate company) in Germany and other countries. All other product and service names mentioned are the trademarks of their respective companies.

Please see <https://www.sap.com/about/legal/trademark.html> for additional trademark information and notices.