



**PUBLIC**

SAP Jam Collaboration

Document Version: 1.1 – 2022-03-08

# SAP Jam Collaboration Developer Guide

# Content

<b>1</b>	<b>Introduction.</b>	<b>4</b>
1.1	The SAP Jam Collaboration Developer Guide.	4
<b>2</b>	<b>External Applications.</b>	<b>7</b>
2.1	Integrate new business records.	7
	Prepare the external application or platform.	9
	Add a Trusted Certificate Authority.	11
	Configure SAP Jam Collaboration as a SAML Local Identity Provider.	13
	Add an OAuth Client.	14
	Add a SAML Trusted IDP.	16
	Register the external application in SAP Jam Collaboration.	19
	Develop an OData annotations file to display business records.	23
	Register the business records in SAP Jam Collaboration.	59
	Configure a business record filter.	62
	Configure a business record sort order.	63
	Configuring access to business records in SAP Jam Collaboration.	64
	Manage "external objects" (business records) using the SAP Jam Collaboration API.	65
<b>3</b>	<b>Branding.</b>	<b>68</b>
3.1	Custom Headers.	68
<b>4</b>	<b>OpenSocial Gadgets.</b>	<b>70</b>
4.1	Develop SAP Jam OpenSocial gadgets.	70
	Gadget Contexts.	72
	Gadget Content Types.	73
	Register your gadget with SAP Jam Collaboration.	74
	OpenSocial Gadget XML Structure.	85
	SAP Jam Collaboration OpenSocial Tutorials.	87
	Using OpenSocial Gadgets to extend SAP Jam Collaboration.	131
	SAP Jam OpenSocial JavaScript API reference.	138
<b>5</b>	<b>Integrate Webhooks.</b>	<b>189</b>
5.1	Push notifications for webhooks.	189
	Webhooks - Alias Users.	189
	Webhooks - Alias Users - WAR file.	194
	Webhooks - Groups.	199
	Webhooks - Groups - WAR file.	204
<b>6</b>	<b>UI5 Embeddable Widgets.</b>	<b>210</b>

6.1	UI5 Embeddable knowledge base widget. . . . .	210
<b>7</b>	<b>SAP Jam embeddable widgets. . . . .</b>	<b>215</b>
7.1	About SAP Jam embeddable widgets. . . . .	215
	SAP Jam Collaboration div-embedded widgets. . . . .	217
<b>8</b>	<b>SAP Jam API. . . . .</b>	<b>233</b>
8.1	About the SAP Jam API. . . . .	233
	SAP Jam API Reference. . . . .	233
	SAP Jam OData API Tutorial. . . . .	234
	Authentication and Authorization API. . . . .	395
	SAP Jam REST API. . . . .	436
<b>9</b>	<b>SAP BTP. . . . .</b>	<b>447</b>

# 1 Introduction

## 1.1 The SAP Jam Collaboration Developer Guide

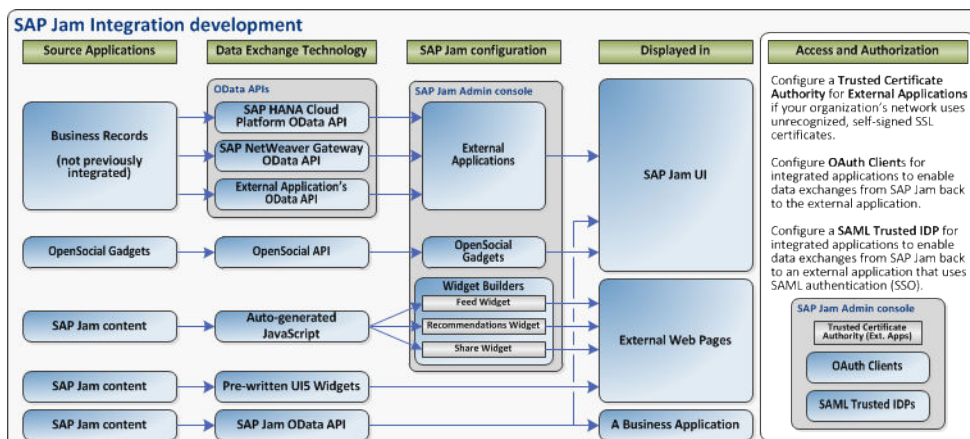
This document, the *SAP Jam Collaboration Developer Guide*, presents developer-oriented information for integrating SAP Jam with other business-critical applications to enable work patterns. Work patterns are repeatable business processes that bring together people's expertise, content, tools, and business data, in a clearly laid out, purpose-built, collaborative workspace—SAP Jam groups—to efficiently and effectively deliver business results. Work patterns combine the collaborative features and tools of SAP Jam with external data, applications, and business best practices. It is the goal of the SAP Jam Developer Community to support customers and partners in building their own SAP Jam work patterns that can solve their specific business problems. While each work pattern is unique, they are built with a common set of building blocks, which include:

- **SAP Jam Group Templates:** SAP Jam groups act as the collaborative workspaces for work patterns. Group templates organize the content in a way that provides guidance to users on the best way to approach the work required to support a particular repeatable business process. The available SAP Jam group templates, and the use and customization of group templates, is discussed in the [SAP Jam Collaboration Administrator Guide](#).
- **Incorporating External Content into SAP Jam:** Incorporating external content into SAP Jam allows data from both other SAP business systems and third party business systems to be displayed within SAP Jam groups. The procedure for doing this involves the following steps:
  - **Register access to the External Application:** This is done by configuring the access in SAP Jam's Admin console [External Applications](#) section, which is explained in the External Application section of the [SAP Jam Collaboration Administrator Guide](#).
  - **Develop the display component for the imported data:** This is done by your organization's developer/integrators developing the components that display data retrieved from these external applications' OData APIs using SAP Jam Annotations, which is documented in the [External Application \[page 7\]](#) section of this *SAP Jam Collaboration Developer Guide*.

These External Data objects can then be bound to specific group templates to allow organizations to easily build groups designed for collaboration around that external data.

## SAP Jam integrations

The integrations of external applications with SAP Jam that are of interest to SAP Jam developer/integrators are shown in the following diagram.



SAP Jam integrations

In summary, there are four major sections in this *SAP Jam Collaboration Developer Guide*:

- **External Applications: [page 7]** This section contains information on integrating business records data from business-critical applications into SAP Jam. This is done by pulling in data from the business application's OData API, or from an intermediary platform that provides an OData API (such as SAP NetWeaver Gateway or SAP BTP), creating an annotations file that maps that data into UI elements of SAP Jam's Business Record Viewer, and registering the external application with SAP Jam (in the Admin console's *External Applications* section).
- **OpenSocial Gadgets: [page 70]** provides documentation on the development of web apps that utilize SAP Jam's implementation of the OpenSocial specification, which extends the capabilities of SAP Jam to interact with any software or service. This section includes a tutorial, a reference section detailing SAP Jam's OpenSocial APIs, and information on configuring these gadgets for use in SAP Jam via the Admin console's *OpenSocial Gadgets* section.
- **Embeddable Widgets: [page 215]** This section describes two mechanisms for adding SAP Jam content to external web pages:
  - **SAP Jam div-embedded widgets [page 217]:** use the SAP Jam Admin console's *Widget Builders* section to generate the JavaScript required to paste a customized *Feed Widget*, *Recommendations Widget*, or *Share Widget* into an HTML page, without requiring knowledge of JavaScript.
- **SAP Jam API: [page 233]** provides the following sections of API Documentation for incorporating SAP Jam content into external applications, or for manipulating SAP Jam via the API:
  - **SAP Jam API Reference [page 233]** has been relocated to <https://jam2.sapjam.com/ODataDocs/ui>.
  - **SAP Jam OData API Tutorial: [page 234]** provides a guided introduction to using the SAP Jam API to integrate SAP Jam content into your business-critical applications, and data from your business critical applications into SAP Jam.
  - **SAP Jam Authentication and Authorization API: [page 395]** provides documentation on the use of SAP Jam's API calls for *OAuth1.0a 3-Legged workflow*, *SAML assertions*, and *single-use tokens* to provide secure access the SAP Jam's OData and REST API calls.
  - **SAP Jam REST API: [page 436]** provides documentation on SAP Jam's legacy REST API calls, which includes Social Reports functions and Memberlist functions.

## Note

An important part of any integration is ensuring that access and authorization are securely and effectively handled, which may require that you:

- **Add a Trusted Certificate Authority [page 11]**

- [Add an OAuth Client \[page 14\]](#)
- [Add a SAML Trusted IDP \[page 16\]](#)

## 2 External Applications

### 2.1 Integrate new business records

There are two types of major applications whose content can be integrated into SAP Jam Collaboration in the [Admin > External Applications](#) section:

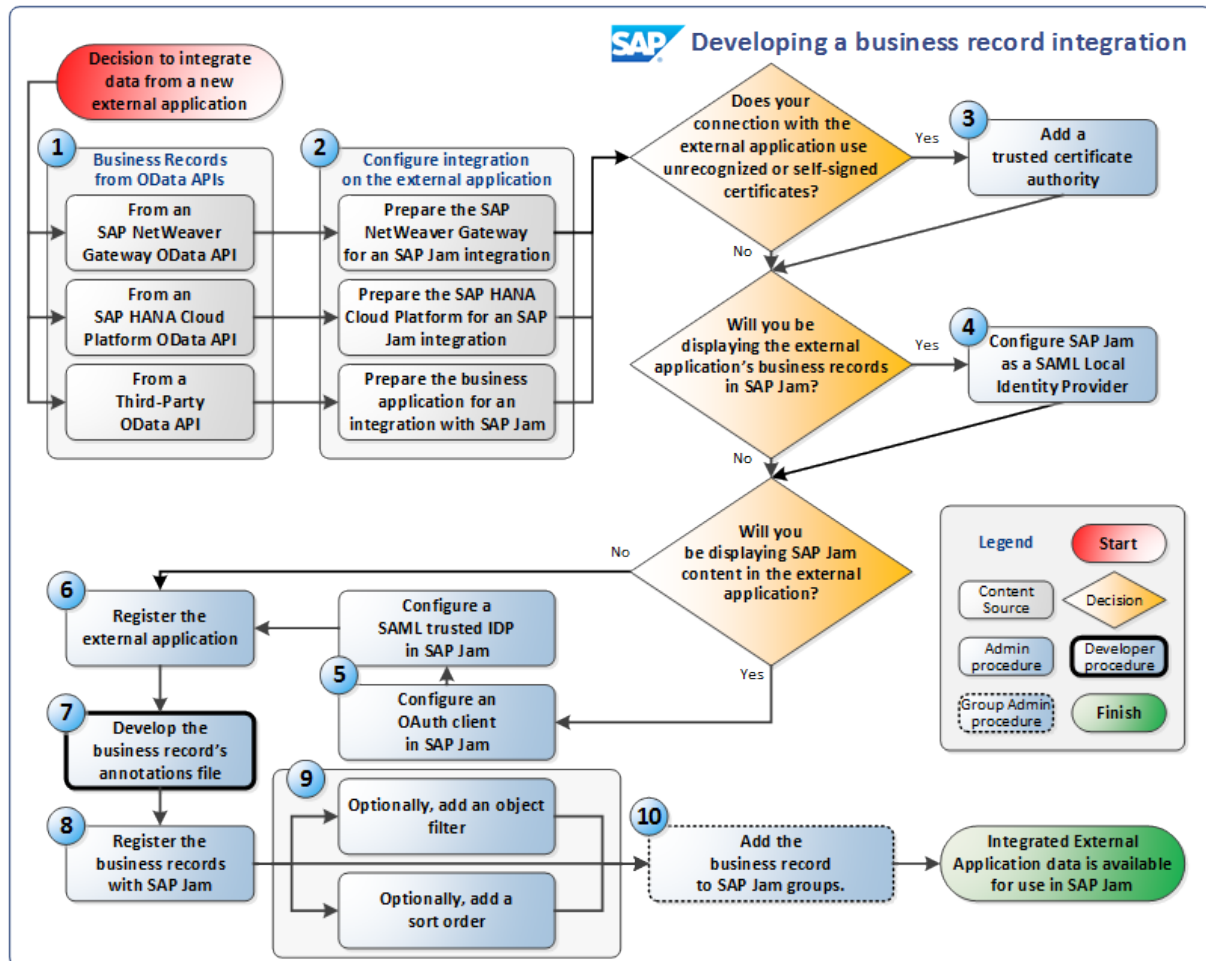
- **Document Repositories:** For more information on integrating documents from an external document repository or library, please see [Integrate document repositories](#) in the *SAP Jam Collaboration Administrator Guide*.
- **Business Records:** The integrations that appear in the [Integrate business records](#) section of the *SAP Jam Collaboration Administrator Guide* are "pre-packaged" integrations, in which the URLs for the external application's OData metadata files and annotation files are listed, and the annotations files are fully developed for the display of certain types of data from the external applications.

To integrate data from a new application, or data that is not currently accessible from a pre-packaged integration, then you must work through the integration steps that are outlined in this section.



## Procedures for integrating new business records into SAP Jam

The steps for integrating new business records are shown in the following diagram, in the procedure following this diagram, and in the pages of this section of the *SAP Jam Developer Guide*:



Work required to develop a new business records integration

- Decide which "application type" that you are going to use for your integration:
  - SAP BTP:** provides a cloud-based platform for your integration, and its services include the ability to add an OData API for applications that do not have one.
  - SAP NetWeaver Gateway:** provides a non-cloud (on-premises network service) platform for your integration, and its services also include the ability to add an OData API for applications that do not have one.
  - Third-Party OData API:** allows you to directly use the OData API of applications that already have them.
- [Configure the external application, and/or its intermediary enabling platform \[page 9\]](#), to allow SAP Jam to access the business application's data using secure authorization mechanisms.
- [Add a Trusted Certificate Authority \[page 11\]](#), if your organization's network requires the use of self-signed TLS (SSL) certificates.
- [Configure SAP Jam as a SAML Local Identity Provider \[page 13\]](#) if you want to display the external application's business records in SAP Jam. This step ensures that users can view only the content from the external application that they have been authorized to view when that material is displayed in SAP Jam.



5. If you want to display SAP Jam content in the external application, you must:
  - [Add an OAuth Client \[page 14\]](#). This configuration provides the external application with authorized access to the SAP Jam API.
  - [Add a SAML Trusted IDP \[page 16\]](#). This step ensures that users can view only the content from SAP Jam that they have been authorized to view when that material is displayed in the external application.
6. [Register the external business application \[page 19\]](#) in SAP Jam's **Admin > External Applications** section, which involves setting the URLs to the application's OData \$metadata file and configuring the appropriate authorization mechanisms between the two.
7. [Develop an annotation file to display the business record \[page 23\]](#). The annotation file is a file that maps the data from the external application to specified UI elements that can be viewed in SAP Jam using its "business record viewer".
8. [Register each of the annotations files \(business records\) \[page 59\]](#) that you developed in the previous step, and that you want to be able to view in SAP Jam.
9. Optionally, you can modify how the business records information is displayed in SAP Jam by doing the following:
  - [Add a business object filter \[page 62\]](#) to show only a select subset of the data for a business record in each configured instance of the business record.
  - [Configure a business record sort order \[page 63\]](#) to control the order in which the business records are displayed in SAP Jam.
10. Finally, as a group administrator, [configure access to the business records in SAP Jam Groups \[page 64\]](#). Alternatively, you can [manage external objects \(business records\) using the SAP Jam API \[page 65\]](#).

The following pages in this section will describe the configuration work required on the external application, on any intermediary platform, and on SAP Jam, and they will show sample OData annotations files and walk you through their development. Additional support is available from the [SAP Jam Developer Community](#).

#### i Note

**Important:** Note that there are two sub-pages of the "Develop an OData annotations file to display business records" page that are important as prerequisites and debugging information in the development of your integration:

- [OData Assumptions \[page 41\]](#) describes the prerequisites for your external application's OData API, particularly for developing your annotations file.
- [External Application Error Messages \[page 49\]](#) describes the types of error messages that you may encounter at various stages of developing your external business records application integration, although again these error messages will largely be encountered when testing out your annotations file.

#### i Note

**Next:** If you are ready to begin your new integration of an external business application with SAP Jam, and you have decided which application type you are going to use, you must [Prepare the external application or platform \[page 9\]](#).

## 2.1.1 Prepare the external application or platform

Once you have decided which application type that your integration will use, you must configure the external application, and/or its intermediary enabling platform, to allow SAP Jam Collaboration to access the business

application's data using secure authorization mechanisms. While this configuration depends on which application type you have selected for your integration, there are some basic requirements that these configurations must meet.

The external application must be configured to enable the following:

- It must allow SAP Jam to access its OData API \$metadata file.
- It must allow SAP Jam to access its OData API calls.
- All of the external application's OData API must:
  - Be resolvable on the public internet
  - Return valid XML
  - Support `$filter` query operations
  - Be compatible with the different caching periods that SAP Jam provides:
    - Results from metadata are cached for 15 minutes
    - Annotation files are cached for 15 minutes
    - XML documents are cached for 3 minutes


For more information on these requirements, see the [OData Assumptions \[page 41\]](#) page.

## Configure an SAP BTP integration

SAP BTP offers a wide array of options for accessing an application's data via an OData API. To select and configure the option that works best for you and your organization, please see the SAP BTP documentation Help Portal at <https://help.hana.ondemand.com/>.


## Configure an SAP NetWeaver Gateway integration

Consult the following documents to configure your SAP NetWeaver Gateway integration:

- [Using OAuth 2.0 from a Web Application with SAML Bearer Assertion Flow](#) 
- [Distributing SAP Gateway Notifications to SAP Jam Collaboration](#)
- [Connecting SAP Jam with SAP ABAP Systems](#)

## Configure a Third-Party OData API integration

There is no specific guide for a third-party OData API integration. You must consult the application's documentation for instructions on how to proceed. However, these document links are provided:

- As a guide to enabling SAML authorization for access to your third-party OData API, see [Using OAuth 2.0 from a Web Application with SAML Bearer Assertion Flow](#) .
- As an example of a successful integration with another application's OData API integration with SAP Jam, see [Integrate SAP Cloud for Customer and SAP Jam](#).

## i Note

**Next:** This page dealt with preparing the external application for integration with SAP Jam Collaboration. You must also perform the required access and authorization configuration procedures required in SAP Jam:

- If your organization's network requires the use of self-signed or unrecognized TLS or SSL certificates, you must [Add a Trusted Certificate Authority \[page 11\]](#).
- If you want to display the external application's business records in SAP Jam, you must [Configure SAP Jam as a SAML Local Identity Provider \[page 13\]](#). This step ensures that users can view only the content from the external application that they have been authorized to view when that material is displayed in SAP Jam.
- If you want to display SAP Jam content in the external application, you must:
  - [Add an OAuth Client \[page 14\]](#). This configuration provides the external application with authorized access to the SAP Jam API.
  - [Add a SAML Trusted IDP \[page 16\]](#). This step ensures that users can view only the content from SAP Jam that they have been authorized to view when that material is displayed in the external application.

## 2.1.2 Add a Trusted Certificate Authority

If your organization's network uses unrecognized or self-signed certificates, then you must perform the following steps for SAP Jam Collaboration to accept the certificate, which will then allow SAP Jam Collaboration access to the external application data.

Note that this can also be a good way to establish a secure connection to your document repositories via CMIS.

## i Note

Adding a trusted certificate authority disables the use of certificates in the default Certificate Authority store. Therefore, if you add a trusted certificate authority, you must manually add all required Certificate Authorities in this section of the Admin console.

1. To find the certificate to add to SAP Jam, you must open a page of the external application that you are trying to integrate into SAP Jam in your browser and view the certificate.  
The way that you do this will vary with each browser, so you must refer to your browser's documentation for instructions.
2. Export the certificate into **Base-64 encoded X.509 (.cer)** format.
3. Open the exported certificate in a text editor and copy the entire certificate, including the "Begin Certificate" and "End Certificate" lines.
4. Open the SAP Jam Admin console, select **Integrations > External Applications** from the left navigation sidebar, and click **Trusted Certificate Authorities** at the top of the **External Applications** page.  
The **Trusted Certificate Authorities** dialog box is displayed.
5. Paste the copied certificate into the **Trusted Certificate Authorities** text box, and click **Submit**.  
This registers the unrecognized certificate with SAP Jam, which will enable the connection to your external application.

### Trusted Certificate Authorities

-----BEGIN CERTIFICATE-----

-----END CERTIFICATE-----

-----BEGIN CERTIFICATE-----

Submit

Cancel

Certificate:

Data:

```

Version: 3 (0x2)
Serial Number: 16777216 (0x1000000)
Signature Algorithm: sha1WithRSAEncryption
Issuer: C=DE, O=SAP-AG, OU=SAPNet, CN=
Validity
  Not Before: May  4 11:56:34 1998 GMT
  Not After : Jul 18 12:00:00 2015 GMT
Subject: C=DE, O=SAP-AG, OU=SAPNet, CN=
Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  RSA Public Key: (1024 bit)
  Modulus (1024 bit):
    00:df:93:4a:e8:c9:a6:ac:22:ef:ba:be:01:0e:1b:
    a0:15:ed:85:51:fc:09:96:f1:2b:69:f0:5b:7e:f9:
    fe:9a:49:fc:8f:10:bf:2a:5d:2d:ad:3b:10:00:32:
    73:61:49:c2:74:32:f2:4e:84:4b:9e:cd:3e:71:dc:
  
```

The add Trusted Certificate Authorities form

## i Note

**Next:** This page dealt with configuring a trusted certificate authority to provide access to external services that use self-signed or unrecognized TLS or SSL certificates. You must also perform any other required authorization configuration procedures required in SAP Jam:

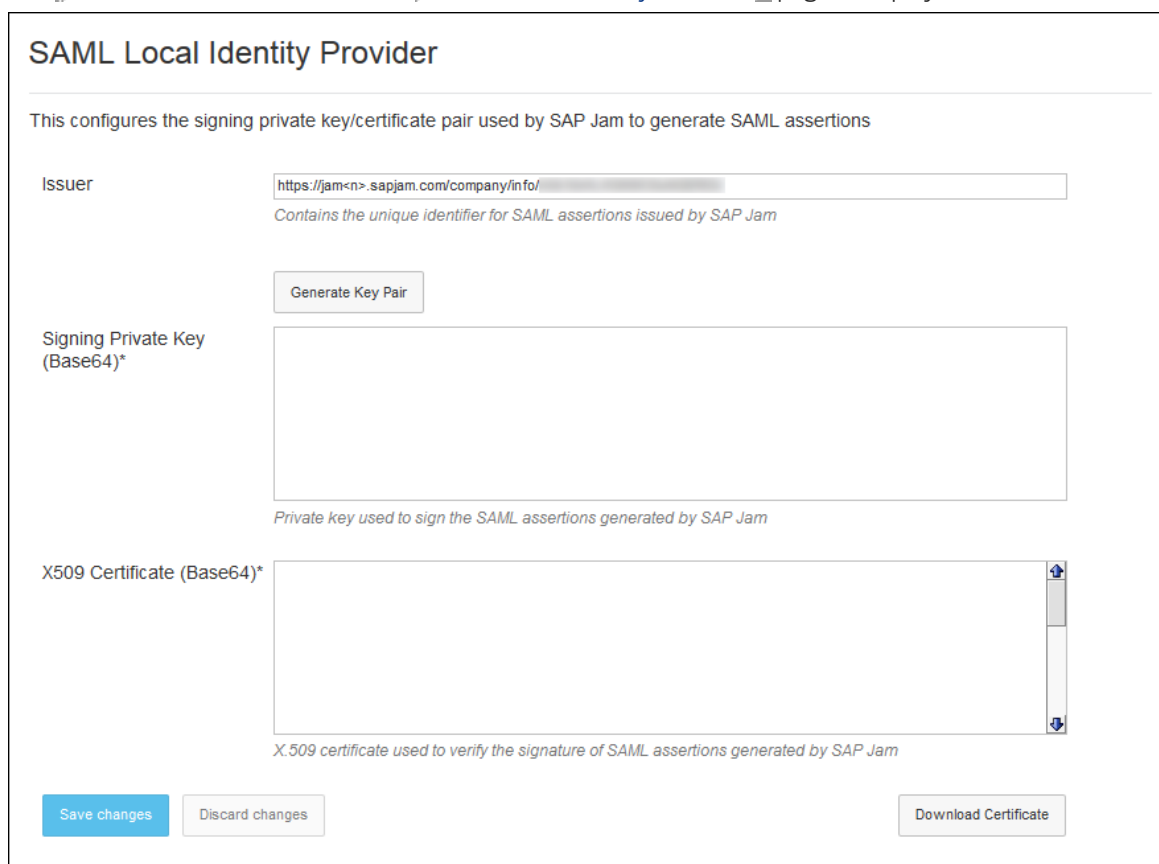
- If you want to display the external application's business records in SAP Jam, you must [Configure SAP Jam as a SAML Local Identity Provider \[page 13\]](#). This step ensures that users can view only the content from the external application that they have been authorized to view when that material is displayed in SAP Jam.
- If you want to display SAP Jam content in the external application, you must:
  - [Add an OAuth Client \[page 14\]](#). This configuration provides the external application with authorized access to the SAP Jam API.
  - [Add a SAML Trusted IDP \[page 16\]](#). This step ensures that users can view only the content from SAP Jam that they have been authorized to view when that material is displayed in the external application.

## 2.1.3 Configure SAP Jam Collaboration as a SAML Local Identity Provider

This procedure configures SAP Jam to allow it to access an external application's user ID and authorization information so that it will display only the material that a user is authorized to view when they view pages where integrated external application business records are displayed in SAP Jam.

To configure SAP Jam as a SAML identity provider, do the following:

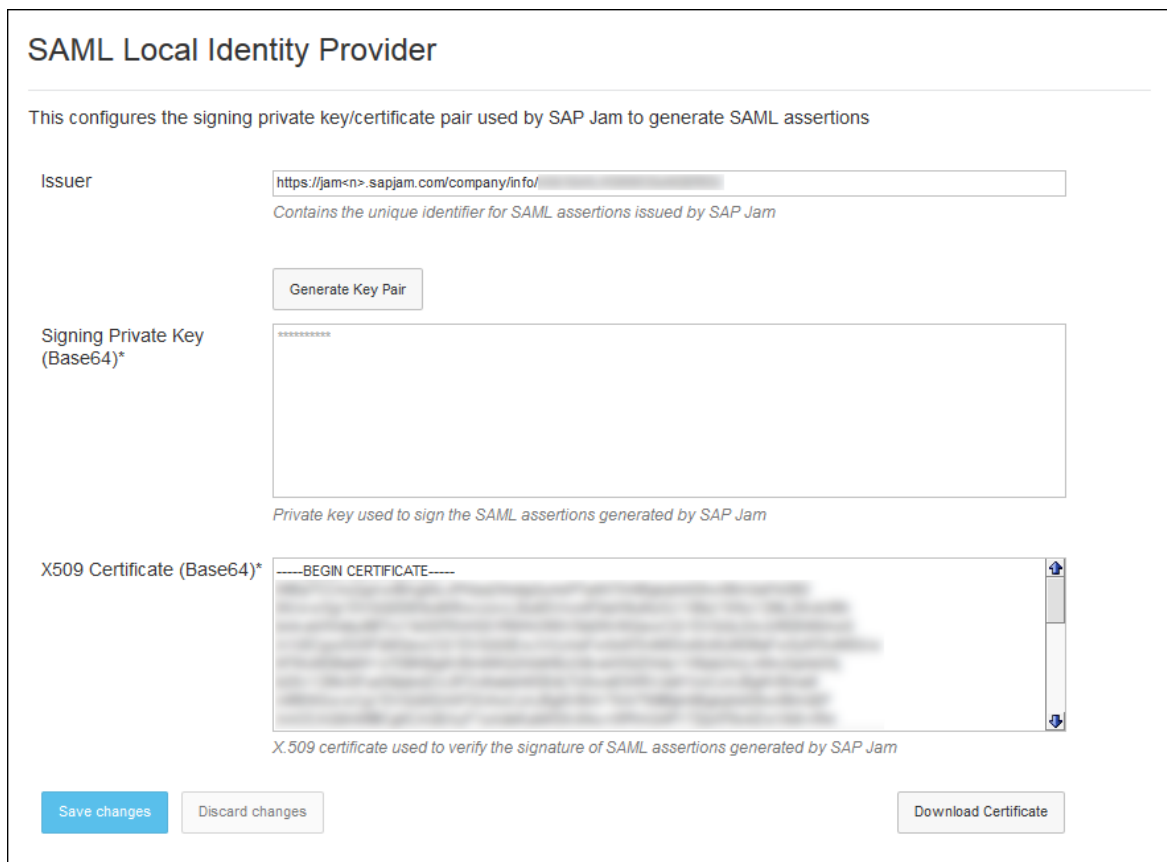
1. In SAP Jam, as a company administrator, click on the cog icon at the top of the page and select [Admin](#) from the context menu.  
The SAP Jam [Admin](#) page is displayed.
2. From the [Admin](#) sidebar menu, select [SAML Trusted IDPs](#).  
The [SAP Jam Collaboration Admin](#) > [SAML Local Identity Provider](#) page is displayed.



SAML Local Identity Provider page, blank

The [Issuer](#) field contains the URI that shows the issuer's identity, in this case, your company's SAP Jam instance.

3. Click [Generate Key Pair](#) to automatically fill the [Signing Private Key](#) and [X509 Certificate](#) text boxes.



**SAML Local Identity Provider**

This configures the signing private key/certificate pair used by SAP Jam to generate SAML assertions

**Issuer**   
*Contains the unique identifier for SAML assertions issued by SAP Jam*

**Signing Private Key (Base64)\***

*Private key used to sign the SAML assertions generated by SAP Jam*

**X509 Certificate (Base64)\***   
*X.509 certificate used to verify the signature of SAML assertions generated by SAP Jam*

SAML Local Identity Provider page, filled

The two text boxes are automatically filled with the signing private key and the X509 certificate (in base64 format), that are required.

4. Copy the information in the *Issuer* and *X509 Certificate* fields to configure SAP Jam as the SAML identity provider in an external application, which provides SAP Jam users with SSO access to the external applications.

### Note

**Next:** This page dealt with configuring SAP Jam as a local SAML identity provider, which ensures that users can view only the content from the external application that they have been authorized to view when that material is displayed in SAP Jam. If you also want to display SAP Jam content in the external application, you must:

- [Add an OAuth Client \[page 14\]](#). This configuration provides the external application with authorized access to the SAP Jam API.
- [Add a SAML Trusted IDP \[page 16\]](#). This step ensures that users can view only the content from SAP Jam that they have been authorized to view when that material is displayed in the external application.

## 2.1.4 Add an OAuth Client

You can authorize an external application to access the SAP Jam API by registering the application as an OAuth client.

## To manage OAuth Clients

1. Go to the SAP Jam Admin console and select **Integrations > OAuth Clients** from the left side navigation.

OAuth Clients			
OAuth clients registered for your company.			
			<a href="#">Add OAuth Client</a>
Name	Integration URL	Date Added	
Example.com's CRM	http://sap-crm.example.com	February 27, 2015 11:46 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's ECC (SD)	https://sap-ecc-sd.example.com	April 09, 2015 02:27 PM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's C4C	https://sap-c4c.example.com	August 10, 2015 11:25 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>

### OAuth Clients catalog

This page presents a catalog of previously configured OAuth Clients, with UI controls that allow you to [View](#), [Edit](#), or [Delete](#) existing OAuth Clients, or to add a new OAuth client ([Add OAuth Client](#)).

2. To add an OAuth client, click [Add OAuth Client](#) at the upper right corner of the page. The [Register a new OAuth Client](#) page displays.
  - In the [Name](#) field, enter a meaningful name that allows company administrators to recognize what the client is.
  - (Optional) From the [Feed Filtering](#) dropdown menu, select either [none](#) or [SAP CRM](#).
  - In the [Integration URL](#) field, enter the URL to the client application API metadata.
  - (Optional) In the [Callback URL](#) field, enter the callback URL for the client application API calls.
  - (Optional) In the [Support URL](#) field, enter the support URL for the client application API.
  - (Optional) Select the [Can Suppress Notifications](#) checkbox to allow the suppression of notifications from external data sources that use this OAuth client. It is up to the developer of this external application integration whether they disable notifications or not, but this setting determines whether notification suppression is permitted from this external application.
  - (Optional) Select the [Can Suppress Webhooks](#) to allow the suppression of webhooks for specific OData calls in the OAuth client. It is up to the developer of this external application integration whether they disable webhooks or not, but this setting determines whether webhook suppression is permitted from this external application.
  - (Optional) In the [X509 Certificate \(Base64\)](#) text box, enter the Transport Layer Security (TLS; supersedes SSL) public key certificate string for the client application API access.
  - (Optional) The [Administrative Area](#) dropdown menu allows you to select the area in which you want this OAuth Client configuration to be available. The default is "Company", which makes it available to all groups and areas. Selecting a specific area limits the scope of the OAuth Client configuration and limits the management of that configuration to either area administrators assigned responsibility for that area or to company administrators.
  - When all of the above settings are complete, click [Save](#) to save the record and establish the trust relationship with the OAuth client application.You are returned to the [OAuth Clients](#) page, with the OAuth client record that you just added listed in the catalog.



3. To view the information for an OAuth client, click [View](#) on the row for the OAuth Client that you want to view.  
The *OAuth Client: <OAuth\_client\_name>* page displays.  
You can either modify the information by clicking [Edit](#) or return to the *OAuth Clients* page by clicking [Back](#).
4. To edit an OAuth client record, either click [Edit](#) in the *OAuth Clients* page or in the *OAuth Client: <OAuth\_client\_name>* page.  
The *Edit your OAuth Client* page displays, which is effectively identical to the *Register a new OAuth Client* page.
  1. Make whatever changes are required.
  2. Click [Save](#) to save your changes.  
You are returned to the *OAuth Clients* page, with the modified OAuth client record that you just edited listed in the catalog.

## 2.1.5 Add a SAML Trusted IDP

By registering an external application as a SAML trusted Identity Provider (IDP), you allow the application to access SAP Jam user ID and authorization information. Users then see only the content they are authorized to view when SAP Jam features are integrated into the external application.

### To manage SAML Trusted IDPs

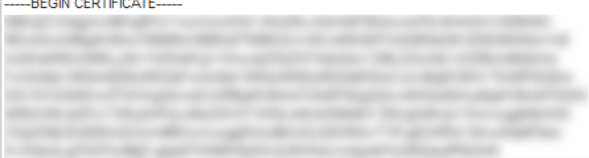
1. Go to the SAP Jam Admin console and select [Integrations](#) [SAML Trusted IDPs](#) from the left side navigation.

SAML Trusted IDPs		
Company IDPs		
SAML Trusted Identity Providers registered for the entire company.		Register your SAML Trusted IDP
IDP ID	Date Added	
<input checked="" type="checkbox"/> QI3_004	October 31, 2012 02:16 AM	Action ▾
<input checked="" type="checkbox"/> U3Y/001	October 31, 2012 08:35 AM	Action ▾
<input checked="" type="checkbox"/> UI3_000	October 31, 2012 08:36 AM	Action ▾
<input checked="" type="checkbox"/> Q63_002	November 06, 2012 06:03 AM	Action ▾

SAML Trusted IDPs catalog

2. To add an identity provider, click [Register your SAML Trusted IDP](#) near the upper right corner of the page.

## Register a new SAML Trusted Identity Provider

Metadata File	<input type="text"/>	<input type="button" value="Browse"/>
	<i>Configure Single Sign-On settings from a SAML metadata file issued by the IDP</i>	
IDP ID*	<input type="text" value="https://sapediad066389trial.hanatrial.ondemand.com"/>	
	<i>Issuer of the SAML assertion. This should be provided by the IDP</i>	
Single Sign-On URL	<input type="text"/>	
	<i>Jam uses this URL for SSO with the IDP.</i>	
Single Log-Out URL	<input type="text"/>	
	<i>Jam uses this URL for SLO with the IDP.</i>	
Default Name ID Format	<input type="text"/>	
	<i>Specifies the default Name ID format to use in an authentication request.</i>	
Default Name ID Policy SP Name Qualifier	<input type="text" value="sapediaHCP"/>	
	<i>Specifies the default SP name qualifier to use in an authentication request.</i>	
X509 Certificate (Base64)*	<div> <div>-----BEGIN CERTIFICATE-----</div> <div></div> <div>-----END CERTIFICATE-----</div> </div>	
	<i>x.509 certificate used to verify the signature of assertions supplied by the IDP.</i>	
Enabled	<input checked="" type="checkbox"/>	
	<i>Specifies whether SAML Assertions will be accepted from this IDP</i>	
Administrative Area	<input type="button" value="Company ▼"/>	
	<i>The Administrative Area where this IDP configuration is available.</i>	
Primary	<input type="checkbox"/>	
	<i>Default IDP used for SSO redirect. There can only be one primary IDP in a company.</i>	
<input type="button" value="Register"/>		

**Register a new SAML Trusted Identity Provider form**

- Optionally, beside the [Metadata File](#) field, click [Browse](#). Search for the metadata file issued by your SAML identity provider on your local hard drive and upload it with your browser file upload feature. May fields on the form are automatically filled.
- In the [IDP ID](#) field, enter the URL of your identity provider, or a name that indicates who the SAML trusted ID provider is, or the application name for which the SAML trusted ID provider is providing single sign-on services.
- In the [Single Sign-On URL](#) field, enter the URL used for single sign-on (SSO) with the identity provider (IDP).
- In the [Single Log-Out URL](#) field, enter the URL used for single log-out (SLO) with the identity provider (IDP).
- In the [Default Name ID Format](#) field, enter what name ID format is to be used in an authentication request.
- In the [Default Name ID Policy SP Name Qualifier](#) field, enter the default Service Provider (SP) name qualifier that is to be used in an authentication request.

- In the *X509 Certificate (Base64)* text box, enter the Transport Layer Security (TLS) public key certificate string for the client application's API access.
- Optionally, select the *Enabled* checkbox to make this SAML trusted IDP immediately available. You can enable or disable the entry in the SAML Trusted IDPs page at any time.
- The *Administrative Area* dropdown menu allows you to select the area in which you want this SAML Trusted IDP configuration to be available. The default is "Company", which makes it available to all groups and areas. Selecting a specific area limits the scope of the SAML Trusted IDP configuration and limits the management of that configuration to either area administrators assigned responsibility for that area or to company administrators.
- Select the *Primary* checkbox to indicate that this SAML trusted identity provider should be the preferred provider. A secondary provider can be specified, which can be set to ensure access should the primary provider fail.

### **i** Note

There can be only one primary IDP configured within your organization.

- Select whether the SAML Issuer ID is globally unique.
  - Select "Company Specific" if:
    - Your organization uses a custom subdomain; or
    - Your organization's IdP is connected to other SAP Jam instances.
  - Select "Global" if:
    - Your organization is using the SAIL or SMI libraries for integration; or
    - Your organization does not use a custom subdomain.
- When all of the above settings are complete, click *Save* to save the record and establish the trust relationship with the SAML Trusted IDP. You are returned to the *SAML Trusted IDPs* page, with the SAML Trusted Identity Provider record that you just added listed in the catalog.
- To enable or disable a SAML Trusted IDP, change the slider control in the left-most column of the row for the SAML Trusted Identity Provider for which you want to change its enabled status.
    - A disabled SAML Trusted IDP appears gray and displays an "X" to the right of the button. Click the control to enable the SAML Trusted IDP.
    - An enabled SAML Trusted IDP appears blue and displays a checkmark to the left of the button. Click the control to disable the SAML Trusted IDP.
  - To view the information for a SAML Trusted Identity Provider, click *Action* in the right-most column of the row for the SAML Trusted Identity Provider that you want to view in the table listing your organization's configured IDPs and select *View* from the menu. The *SAML trusted IDP: <SAML\_trusted\_IDP\_name>* page displays, showing the details of this SAML trusted IDP configuration. You can either modify the information by clicking *Edit* or return to the *SAML Trusted IDPs* page by clicking *Back*.
  - To edit a configured SAML Trusted IDP, either click *Action* in the right-most column of the row for the SAML Trusted Identity Provider that you want to view in the table listing your organization's configured IDPs and select *Edit* from the menu, or click *Edit* in the *SAML trusted IDP: <SAML\_trusted\_IDP\_name>* page. The *Edit your SAML Trusted Identity Provider* page displays, which is effectively identical to the *Register a new SAML Trusted Identity Provider* page.
    1. Make whatever changes are required.
    2. Click *Save* to save your changes.

You are returned to the [SAML Trusted IDPs](#) page, with the modified SAML Trusted Identity Provider record that you just edited listed in the catalog.

- To delete a SAML Trusted Identity Provider entry, click [Action](#) in the right-most column of the row for the SAML Trusted Identity Provider that you want to delete in the table listing your organization's configured IDPs and select [Delete](#) from the menu.

A confirmation dialog box appears. Click [Delete](#) to confirm and remove the selected SAML Trusted IDP record.

You are returned to the [SAML Trusted IDPs](#) page.

## 2.1.6 Register the external application in SAP Jam Collaboration

Create an External Application entry to register your external application with SAP Jam Collaboration. This procedure establishes a secure data connection between SAP Jam and your external business application.

- In SAP Jam, as a company administrator, click on the cog icon at the top of the page and select [Admin](#) from the context menu.

The SAP Jam [Admin](#) console displays.

- Select [Integrations](#) [External Applications](#) from the Admin console navigation sidebar.

The [External Applications](#) page is displayed. This initial page is a catalog of the external applications that have previously been configured for your organization's SAP Jam instance.

- Click [Add Application](#).

A drop-down menu listing the available external application types displays.

**External Applications**

Business records integration options

Application Type	Action
SAP HANA Cloud Platform	Action ▼
SharePoint 2013	Action ▼
SharePoint 2013	Action ▼
Box	Action ▼
SAP NetWeaver Gateway	Action ▼
SAP Hybris Cloud for Customer	Action ▼

### Add an External Application

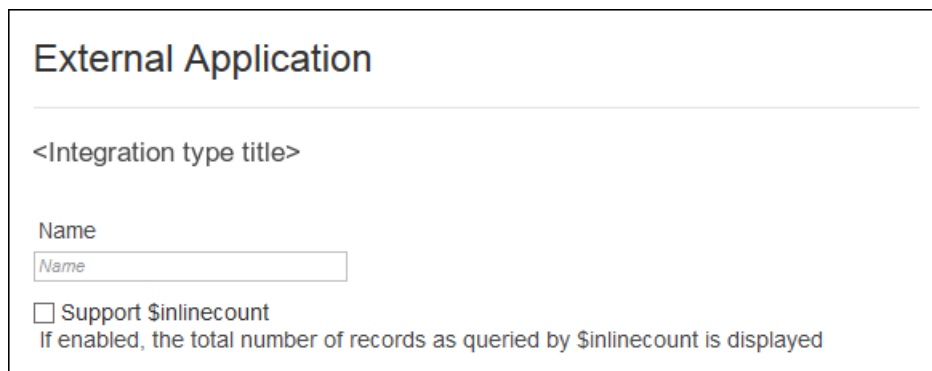
- Select the external application type that you want to add from the drop-down menu.

The options of interest for adding new business records are:

- [SAP BTP](#): to incorporate data from a cloud-based business application via the SAP BTP.
- [SAP NetWeaver Gateway](#): to incorporate data from a on-premises business application via SAP NetWeaver Gateway.
- [Third Party OData Source](#): to incorporate data directly from a business application's OData API.

Once you select an application type, the form for adding a new external application is displayed.

Each of the sections of this form and their options are described in the following:



External Application, first options

- Type in a meaningful name in the [Name](#) text box, such as one that names the external application and/or its dedicated use.

#### **i** Note

This name will appear in the SAP Jam navigation sidebar and in other locations in SAP Jam, so it is important to make it something recognizable.

- **[For NetWeaver and SAP BTP integrations only]** Select the [Support \\$inlinecount](#) check box if you want to display the total number of records of a requested type.  
This is just a total count of the type of records that are being displayed. There will still be a maximum of 10 or 20 items, depending on the type of resource being displayed. (See the "Business Record Editor (BRE)" section in the [OData Assumptions](#) page in this section of the *SAP Jam Developer Guide* for the maximum number of records shown for different types of resources.) For example, with this option enabled, a table displaying customers might show a label below the table stating "Showing items 21-40 of 57 records".

Select the [Authentication Type](#).

For business record integrations, you should only use the [Common user](#) option for testing and development purposes. For production systems, you should use [Per user](#) as the [Authentication Type](#).



External Application, common user authentication options

In [Common User](#) authentication mode, all access from SAP Jam to the external data source will be made through an HTTP Basic authentication call using the provided user credentials. This is recommended only

if you wish to expose all of the data in the data source to all SAP Jam users in your company. In other words, ensure that all the data exposed through the application can be considered "public" within your company.

- **User name:** This is the user name of the generic user account that will be used to access the service on behalf of all of the SAP Jam users.
- **Password:** This is the password of the generic user account that will be used to access the service on behalf of all of the SAP Jam users.

The screenshot shows a configuration window titled "Select Authentication Type". A dropdown menu is set to "Per user". Below this, there are several input fields: "OAuth 2.0 Client Id" (placeholder: OAuth 2.0 Client Id), "Secret" (placeholder: Secret), "Service provider (Protocol, Host, Port)" (with a dropdown set to "https", a text field containing "serviceprovider.com", and a port field set to "443"), "Service Provider Name" (placeholder: Service Provider Name), and "Scope" (placeholder: Scope).

External Application, per-user authentication options

In the *Per-User* authentication mode, SAP Jam authenticates the current user against the external application through the OAuth 2.0 SAML bearer assertion work flow. Since SAP Jam delegates its user authentication to the SuccessFactors BizX platform, a trust relationship must first be established between the external application and the BizX platform. Once this relationship is established, an OAuth client for SAP Jam is configured in the external application, using the BizX IDP as the issuer of identity assertions for the client.

- **OAuth 2.0 Client Id:** Your organization's client ID with your OAuth 2.0 SSO provider.
- **Secret:** Your organization's OAuth 2.0 SSO secret.
- **Service Provider:** The URL of your organization's OAuth 2.0 SSO provider, set in three parts: protocol, host, and port number.
- **Service Provider Name:** The name of your organization's OAuth 2.0 SSO provider.
- **Scope:** Your organization's company name as set in SAP Jam. This can be found in the [Admin](#) [General](#) page.

When an SAP Jam user attempts to access the external data source, SAP Jam will first post a SAML assertion on the user's identity to the external object in exchange for an OAuth token for that user. If successful, SAP Jam will use this OAuth token for all subsequent data access on behalf of that user. More detail can be found at: <http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-20> .

The screenshot shows a single input field labeled "NetWeaver Client Id" with a placeholder text "NetWeaver Client Id".

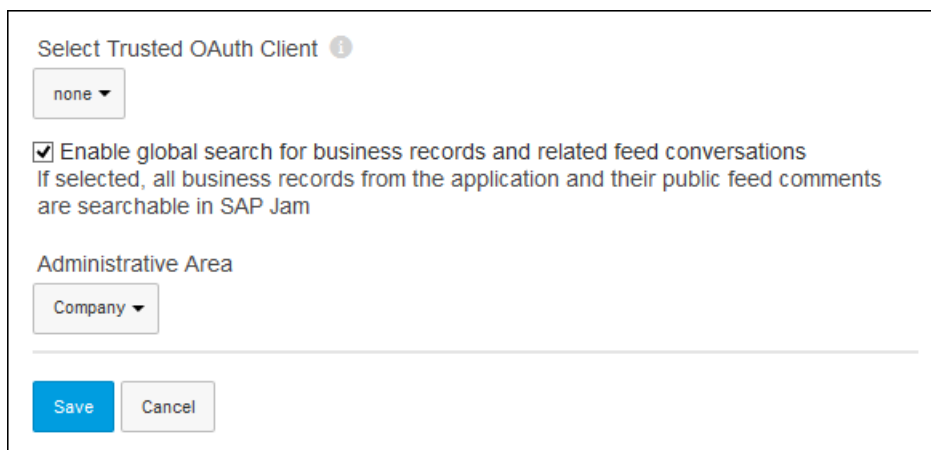
External application, type-specific options: NetWeaver Gateway

- **[SAP NetWeaver Gateway only]** Set the *NetWeaver Client Id*. This is the three-digit login parameter for your NetWeaver client.

☐ Consume as OData V4  
If enabled, the incoming data will be interpreted as OData V4 instead of OData V2

#### External application, type-specific options: Third Party OData Source

- **[Third-Party OData only]** Set the [Consume as OData V4](#) to specify that this external application's API conforms to version 4 of the OData specification; leave this option unchecked to have the external application's OData API interpreted as being version 2 compliant.



#### External Application, last options

- Optionally, select the [Select Trusted OAuth Client](#) from the drop-down menu. This is required if you want to display SAP Jam content in the external application. This configuration provides the external application with authorized access to the SAP Jam API.
- Select [Enable global search for business records and related feed conversations](#) to make all business records from this application, and their related public feed comments, searchable from within SAP Jam.
- The [Administrative Area](#) drop-down menu allows you to select the area in which you want this SAML Trusted IDP configuration to be available. The default is "Company", which will make it available to all groups and areas. Selecting a specific area will limit the scope of the SAML Trusted IDP configuration and will limit the management of that configuration to either area administrators assigned responsibility for that area or to company administrators.
- Click [Save](#).  
You are returned to the [Admin > External Applications](#) page, where the list of external applications that have been added is displayed, now including the external application that you have just registered.

### i Note

#### Next:

This page explained how to create an External Application entry to register your external application with SAP Jam. This procedure is the step that establishes a secure data connection with your external business application, assuming that the required access and authorization procedures have been properly performed on the external application and in SAP Jam. At this point you must now proceed with the development work required to accomplish the objectives of displaying SAP Jam content in your external application or displaying business records from your external application in SAP Jam:

- To display an external application's business records in SAP Jam, you must [Develop an OData annotations file to display business records \[page 23\]](#).



- To display SAP Jam features in an external application, you must develop the GUI components to display the data, and you must pull in the data using SAP Jam API calls. The use of the SAP Jam APIs is documented in the [About the SAP Jam API \[page 233\]](#) section of this *SAP Jam Developer Guide*.

## 2.1.7 Develop an OData annotations file to display business records

The majority of the configuration procedures that you have performed so far have been to establish data communications that either allow SAP Jam Collaboration to access an external business application's OData APIs or that allow SAP Jam to communicate changes that have been made to that data back to the external business application. The data that is exchanged between these services is mostly short strings of text or numbers accessed from an XML or JSON file passed back and forth across your organization's network. To make this data presentable for human consumption, it is mapped into graphical objects using an OData Annotations file, which is organized in accordance with the layout of the graphical user interface (GUI) objects that the SAP Jam team have developed, with indications of what type of data should populate each segment of the GUI.

At this point, a few definitions should be helpful:

- **External Applications:** are applications whose content can be integrated into SAP Jam to be displayed in various locations, or applications that can have SAP Jam content displayed within their screens. Access to most of these applications is configured through the SAP Jam *Admin* console's *External Application* section.
- **Business Records:** is SAP Jam's term for the information from an external application that is incorporated into SAP Jam pages. Integrating external application's Business Records requires the development of an `annotations.xml` file. This development work is a necessary step in configuring access to applications through the SAP BTP, SAP Cloud for Customer, or SAP NetWeaver Gateway.
- **ExternalObjects:** are the SAP Jam OData API term for the UI objects that consume and display the Business Records.

### i Note

**Important:** Note that there are two sub-pages to this page that are important as prerequisites and debugging information in the development of your integration:

- [OData Assumptions \[page 41\]](#) describes the prerequisites for your external application's OData API, particularly for developing your annotations file.
- [External Application Error Messages \[page 49\]](#) describes the types of error messages that you may encounter at various stages of developing your external business records application integration, although again these error messages will largely be encountered when testing out your annotations file.

## SAP Jam GUI objects that display business records

There have been three basic types of GUI objects developed to display Business Records in SAP Jam.

## Record List View

Ace Corp Jam
All ▼
Home Groups ▾ Company Business Objects
Recommendations Bookmarks Calendar

- Feed Updates
- My Bookmarks
- Notifications
- Messages
- Recommendations
- Business Objects

### SugarCRM / SugarCRM Opportunities


Showing: All ▼

Title	Changed At	Sales Stage
Stan Over Trust - 1000 units 2014-05-06 • 50000 • 75 %	2014-05-02 22:48:19	Prospecting
Overhead & Underfoot Ltd. - 1000 units 2015-02-14 • 25000 • 10 %	2014-04-28 18:52:49	Negotiation/Review
Hollywood Diner Ltd - 1000 units 2014-05-24 • 75000 • 40 %	2014-04-28 18:52:49	Proposal/Prico Quote
Super Star Holdings Inc - 1000 units 2015-04-24 • 50000 • 60 %	2014-04-28 18:52:49	Qualification
S Cane Sweeteners Ltd - 1000 units 2014-06-07 • 10000 • 40 %	2014-04-28 18:52:49	Proposal/Prico Quote
Air Safety Inc - 1000 units 2014-06-25 • 25000 • 40 %	2014-04-28 18:52:49	Negotiation/Review
Diamondport Investing - 1000 units 2013-05-23 • 75000 • 40 %	2014-04-28 18:52:49	Closed Lost

Business  
Record Details  
View

A Business Record detailed view is shown below. This will appear as a top-level navigation for groups which are associated directly with an ExternalObject. The details can also be viewed when clicking on a Business Record within a list of Business Records.

Hot Prospects



Super Star Holdings Inc

Featured in 1 Group(s)

Name :

Super Star Holdings Inc

Changed At :

2014-04-28 18:52:49

Account Type :

Customer

Industry :

Technology

Office Phone :

(910) 273-1181

Website :

www.infothe.org

Employee Responsible :

Sally Bronsen

Address :

345 Sugar Blvd.  
52941 Sunnyvale  
USA

Contacts

Opportunities

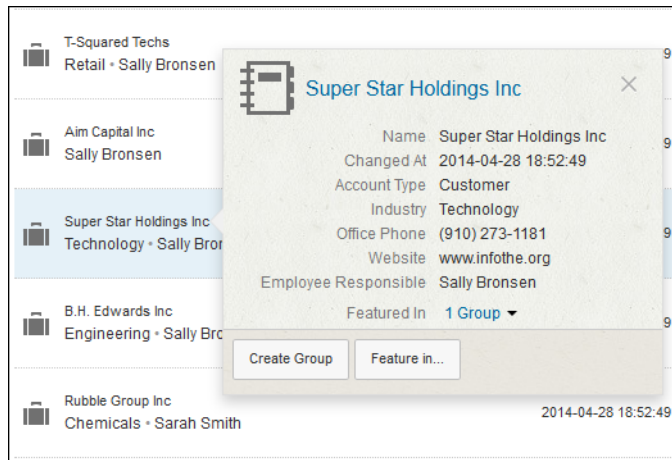
Account Details

Title	Changed At	Sales Stage	Closing Date	Exp. Sales Vol.	Chance of Success %
Super Star Holdings ...	2014-04-28 18:52:49	Qualification	2015-04-24	50000	60 %

## The QuickView Control

The QuickView control renders external object details in a pop-up window when the user hovers their mouse over the object. This control can be active in the SAP Jam feed, in the external object browser, or even when browsing through an object's relationships.

The QuickView also includes several action buttons. The "Create Group" button enables the user to create a new SAP Jam group with the given object as its Primary Object. The "Feature in" button enables the user to associate an external object with an existing group, such that users can build up groups which reference many different external objects.



The fields in the QuickView correspond to the "UI.Identification" annotation term. Each Record object in the UI.Identification collection corresponds to a row in the QuickView details. The values of the properties are bound to the OData properties of the object through the "Path" attribute, as shown.

## About Annotations

An EDMX file is an Entity Data Model XML file, which is made up of a conceptual model, a storage model, and the mapping between these models. An EDMX file also contains information that can be used to render a model graphically. An OData Annotations file leverages this technology to map data from a specified OData API to a layout of that data that can be easily consumed by user interface (UI) component. SAP Jam Annotations, therefore, provide a way to map data from external applications for the graphical display of that data in a page in a SAP Jam group. For more information on OData Vocabularies, which is the basis of OData Annotations, see <http://www.odata.org/vocabularies/>.

An annotations file consists of an EDMX wrapper around an OData Annotations core. The EDMX wrapper describes, and links to, the XML NameSpaces that are used in the mapping. An example of the EDMX parts of an Annotations file, along with the initial top-level Annotations tag, is shown in the following example.

```
<?xml version="1.0" encoding="utf-8"?>
<edmx:Edmx Version="4.0" xmlns:edmx="http://docs.oasis-open.org/odata/ns/edmx">
  <edmx:Reference>
    <edmx:Include Alias="UI" Namespace="com.sap.vocabularies.UI.v1"/>
  </edmx:Reference>
  <edmx:Reference>
    <edmx:Include Alias="Communication"
      Namespace="com.sap.vocabularies.Communication.v1"/>
  </edmx:Reference>
  <edmx:Reference Uri="http://docs.oasis-open.org/odata/odata/v4.0/cs01/
    vocabularies/Org.OData.Measures.V1.xml">
    <edmx:Include Alias="Measures" Namespace="Org.OData.Measures.V1"/>
  </edmx:Reference>
  <edmx:Reference>
    <edmx:Include Alias="SugarCRM" Namespace="com.sap.jam.samples.sugarcrm"/>
  </edmx:Reference>
</edmx:Edmx>
```

```

    <Schema Alias="SugarAnnotation"
Namespace="com.sap.jam.sugarcrm.Annotation"
    xmlns="http://docs.oasis-open.org/odata/ns/edm">
        <Annotations Target="SugarCRM.Opportunity/probability">
            <!-- ... -->
        </Annotations>
    </Schema>
</edm:DataServices>
</edm:Edmx>

```

The core of the annotations file that is *not* shown in the preceding code snippet is the subject of this section.

## Annotations Markup Overview

Some examples of the significant structures of an Annotations file that are discussed in this section are shown in the following code snippet.

```

<Annotations Target="SugarCRM.Contact">
  <Annotation Term="UI.HeaderInfo">
    <Record>
      <PropertyValue Property="TypeName" String="Contact"/>
      <PropertyValue Property="ImageUrl" String="/images/exobj/account/
account_grey_48.png"/>
      <PropertyValue Property="Title">
        <Record>
          <PropertyValue Property="Value" Path="name"/>
        </Record>
      </PropertyValue>
    </Record>
  </Annotation>
  <Annotation Term="UI.Identification">
    <Collection>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="Name"/>
        <PropertyValue Property="Value" Path="full_name"/>
      </Record>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="Title"/>
        <PropertyValue Property="Value" Path="title"/>
      </Record>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="E-Mail"/>
        <PropertyValue Property="Value" Path="email1"/>
      </Record>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="Office Phone"/>
        <PropertyValue Property="Value" Path="phone_work"/>
      </Record>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="Mobile Phone"/>
        <PropertyValue Property="Value" Path="phone_mobile"/>
      </Record>
    </Collection>
  </Annotation>
  <Annotation Term="UI.LineItem">
    <Collection>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="Name"/>
        <PropertyValue Property="Value" Path="full_name"/>
      </Record>
      <Record Type="UI.DataField">
        <PropertyValue Property="Label" String="E-Mail"/>
        <PropertyValue Property="Value" Path="email1"/>
      </Record>
    </Collection>
  </Annotation>

```

```

    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Office Phone"/>
      <PropertyValue Property="Value" Path="phone_work"/>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Mobile Phone"/>
      <PropertyValue Property="Value" Path="phone_mobile"/>
    </Record>
  </Collection>
</Annotation>
</Annotations>

```

The remainder of this page consists of reference documentation for:



- [Annotation Elements \[page 27\]](#), the six main XML elements used in the Annotations core of an SAP Jam annotations file.
- [Term attribute values \[page 33\]](#), the valid values of the Annotation element's Term attribute, which describes the rendered sections of an External Object.
- [Type attribute values \[page 38\]](#), the valid values of the Record element's Type attribute, which specify the UI widget type that is used to present the data mapped in this Record element.
- [Property attribute values \[page 40\]](#), the valid values of the PropertyValue element's Property attribute, which specify the type of data involved, whether that is a UI element or a particular piece of data from the external application's OData API.

## Annotation Elements

### Annotations

**Description:** Describes the data mapping for an External Object.

- **Parent Element:**

Element	Description
Schema	See <a href="#">MSDN &gt; CSDL &gt; Schema Element</a>  and <a href="#">OASIS &gt; edm.xsd</a>  .

- **Attributes:**

Attribute	Use	Description
Target	Mandatory	A reference to an EntityType in the external application's OData API \$metadata.

- **Child Elements:**

Element	Description
<a href="#">Annotation [page 28]</a>	Describes a section of an External Object.

- **Example:** A generalized Annotations element

```

<Schema ...>
  <Annotations Target="[SomeBusinessRecordIdentifier].[EntityName]">

```

```

    <Annotation ...>
    </Annotation>
  </Annotations>
</Schema>

```

## Annotation

**Description:** Describes a the data mapping of a section of an External Object.

- **Parent Element:**

Element	Description
<a href="#">Annotations [page 27]</a>	Describes the data mapping for an External Object.

- **Attributes:**

Attribute	Use	Description
<a href="#">Term [page 33]</a>	Mandatory	Can be one of: <ul style="list-style-type: none"> <li>◦ <a href="#">UI.HeaderInfo [page 34]</a> is the title banner for the External Object.</li> <li>◦ <a href="#">UI.Identification [page 35]</a> contains the identifying details for the External Object.</li> <li>◦ <a href="#">Communication.Address [page 35]</a> presents a mailing address block for use in other UI segments.</li> <li>◦ <a href="#">UI.Facets [page 36]</a> defines a set of tabs in the ExternalObject "details" view.</li> <li>◦ <a href="#">UI.LineItem [page 36]</a> defines a row in a table of items.</li> <li>◦ <a href="#">UI.FieldGroup [page 37]</a> enables the display of a set of fields in a facet that are not in tabular form.</li> <li>◦ <a href="#">Core.Description [page 37]</a></li> <li>◦ <a href="#">UI.IsImageUrl [page 38]</a></li> </ul>
<a href="#">Qualifier</a>	Optional	Qualifies the Term value to use a limited or alternate arrangement of that section of the External Object.

- **Child Elements:**

Element	Description
<a href="#">Collection [page 29]</a>	A container of multiple records. An <i>Annotation</i> can contain a <i>Record</i> or a <i>Collection</i> of <i>Records</i> .
<a href="#">Record [page 29]</a>	A set of related data elements described in PropertyValue elements. An <i>Annotation</i> can contain either a <i>Collection</i> or a <i>Record</i> , but not both.

- **Example 1:** An Annotation element, showing a Term attribute

```

<Annotations ...>
  <Annotation Term="UI.HeaderInfo">
    <Record ...>
    </Record>
  </Annotation>
</Annotations>

```

- **Example 2:** An Annotation element, showing both a Term and a Qualifier attribute

```
<Annotations ...>
  <Annotation Term="UI.FieldGroup" Qualifier="Overview">
    <Record ...>
    </Record>
  </Annotation>
</Annotations>
```

## Collection

**Description:** A container of multiple records.

- **Parent Element:**

Element	Description
<a href="#">Annotation [page 28]</a>	Describes the data mapping of a section of an External Object.

- **Attributes:**

Attribute	Use	Description
None		

- **Child Elements:**

Element	Description
<a href="#">Record [page 29]</a>	A set of related data elements described in child PropertyValue elements.

- **Example:** A Collection element

```
<Annotation ...>
  <Collection>
    <Record ...>
    </Record>
  </Collection>
</Annotation>
```

## Record

**Description:** A set of related data elements described in PropertyValue elements.

- **Parent Element:**

Element	Description
<a href="#">Annotation [page 28]</a>	Describes a the data mapping of a section of an External Object.



Element	Description
<a href="#">Collection [page 29]</a>	A container of multiple records.

- **Attributes:**

Attribute	Use	Description
<a href="#">Type [page 38]</a>	Optional	Can be one of: <ul style="list-style-type: none"> <li>◦ <a href="#">UI.CollectionFacet [page 39]</a></li> <li>◦ <a href="#">UI.DataField [page 39]</a></li> <li>◦ <a href="#">UI.DataFieldForAnnotation [page 39]</a></li> <li>◦ <a href="#">UI.DataFieldWithNavigationPath [page 39]</a></li> <li>◦ <a href="#">UI.DataFieldWithUrl [page 39]</a></li> <li>◦ <a href="#">UI.ReferenceFacet [page 39]</a></li> <li>◦ <a href="#">UI.ReferenceURLFacet</a></li> </ul>

- **Child Elements:**

Element	Description
<a href="#">PropertyValue [page 30]</a>	A Property of a Record. Each Record Type has its own set of mandatory and optional PropertyValue.

- **Example 1:** A Record element, as a child of an Annotation element, and showing a Type="UI.DataField" attribute-value pair

```
<Annotation ...>
  <Record Type="UI.DataField">
    <PropertyValue ...>
    </PropertyValue>
  </Record>
</Annotation>
```

- **Example 2:** A Record element, as a child of a Collection element, and showing a Type="UI.CollectionFacet" attribute-value pair

```
<Collection ...>
  <Record Type="UI.CollectionFacet">
    <PropertyValue ...>
    </PropertyValue>
  </Record>
</Collection>
```

## PropertyValue

**Description:** A single piece of data, which typically can be either data from the source business application, or data related to the UI.

- **Parent Element:**

Element	Description
<a href="#">Record [page 29]</a>	A set of related data elements described in child PropertyValue elements.

- **Attributes:**

Attribute	Use	Description
<a href="#">Property [page 40]</a>	Mandatory	Can be one of: <ul style="list-style-type: none"> <li>◦ <a href="#">ImageUrl [page 40]</a></li> <li>◦ <a href="#">Label [page 40]</a></li> <li>◦ <a href="#">Target [page 40]</a></li> <li>◦ <a href="#">Title [page 40]</a></li> <li>◦ <a href="#">TypeName [page 40]</a></li> <li>◦ <a href="#">TypeNamePlural [page 40]</a></li> <li>◦ <a href="#">Value [page 40]</a></li> </ul>
<b>String</b>	Optional	An alpha-numeric text string. Typically either a UI label string or the URL to a resource such as an image.
<b>Path</b>	Optional	An alpha-numeric text string.
<b>Annotation-Path</b>	Optional	An alpha-numeric text string.
<b>Navigation-Property-Path</b>	Optional	An alpha-numeric text string.

- **Child Elements:**

Element	Description
<a href="#">Annotation [page 28]</a>	Describes a section of an External Object.
<a href="#">Apply [page 32]</a>	Defines an action.
<a href="#">Collection [page 29]</a>	A container of multiple records.
<a href="#">Record [page 29]</a>	A set of related data elements described in PropertyValue elements.

- **Example 1:** A pair of PropertyValue elements with no child elements

```
<Record Type="UI.DataField">
  <PropertyValue Property="Label" String="Supplier ID" />
  <PropertyValue Property="Value" Path="SupplierID" />
</Record>
```

- **Example 2:** A PropertyValue element wrapping an Annotation element

```
<Record ...>
  <PropertyValue Path="@UI.HeaderInfo/ImageUrl" Property="Value">
    <Annotation Term="UI.IsImageUrl"/>
  </PropertyValue>
</Record>
```

- **Example 3:** A PropertyValue element wrapping an Apply element

```
<Record ...>
  <PropertyValue Property="Value">
    <Apply Function="odata.concat">
```

```

        <Path>ContactTitle</Path>
        <String> - </String>
        <Path>ContactName</Path>
    </Apply>
</PropertyValue>
</Record>

```

- **Example 4:** A PropertyValue element, wrapping a Collection element

```

<Record ...>
    <PropertyValue Property="Data">
        <Collection>
            <Record ...>
                <!-- ... -->
            </Record>
        </Collection>
    </PropertyValue>
</Record>

```

- **Example 5:** A PropertyValue element, wrapping a Record element

```

<Record ...>
    <PropertyValue Property="Data">
        <Record ...>
            <!-- ... -->
        </Record>
    </PropertyValue>
</Record>

```

## Apply

**Description:** Defines an action to be performed on specified pieces of data.

- **Parent Element:**

Element	Description
<a href="#">PropertyValue [page 30]</a>	A single piece of data, which typically can be either data from the source business application, or data related to the UI.

- **Attributes:**

Attribute	Use	Description
<b>Function</b>	Mandatory	Valid value is "odata.concat", which concatenates the data indicated in the child Path elements and/or text indicated in the child String elements.

- **Child Elements:**

Element	Description
<b>Path</b>	The URL for the piece of data to be used.
<b>String</b>	The alphanumeric text string that is to be concatenated with the other listed pieces of data and/or strings.

- **Example 1:** An Apply element, showing a concatenation of a ContactTitle and a ContactName

```
<Record ...>
  <PropertyValue Property="Value">
    <Apply Function="odata.concat">
      <Path>ContactTitle</Path>
      <String> - </String>
      <Path>ContactName</Path>
    </Apply>
  </PropertyValue>
</Record>
```

- **Example 2:** An Apply element, showing a concatenation of an employee's FirstName and LastName

```
<Record ...>
  <PropertyValue Property="Value">
    <Apply Function="odata.concat">
      <Path>Employee/FirstName</Path>
      <String> </String>
      <Path>Employee/LastName</Path>
    </Apply>
  </PropertyValue>
</Record>
```

### i Note

The only Annotation elements not documented in this section are the <Path> and <String> elements that can be seen in the code snippets above. Their use should be clear from these examples.

## Term attribute values

**Description:** *Term* is an attribute of the *Annotation* element. The various values of the *Term* attribute define sections of the External Object. Each of the values of *Term* element are described in this section.

For example, in the following screen capture, there are four different *Annotation* element *Term* attribute values demonstrated:

**Hot Prospects**

**Super Star Holdings Inc** Featured in 1 Group(s)

Name : Super Star Holdings Inc  
 Changed At : 2014-04-28 18:52:49  
 Account Type : Customer  
 Industry : Technology  
 Office Phone : (910) 273-1181  
 Website : www.infothe.org  
 Employee Responsible : Sally Bronsen  
 Address : 345 Sugar Blvd.  
 52941 Sunnyvale  
 USA

Contacts Opportunities Account Details

Title	Changed At	Sales Stage	Closing Date	Exp. Sales Vol.	Chance of Success %
Super Star Holdings ...	2014-04-28 18:52:49	Qualification	2015-04-24	50000	60 %

1. **UI.HeaderInfo** is the title banner for the External Object.
2. **UI.Identification** contains the identifying details for the External Object.
  1. **Communication.Address** presents a mailing address block for use in other UI segments.
3. **UI.Facets** defines a set of tabs.
4. **UI.LineItem** defines a row in a table of items.

The code examples for these four sections are as follows:

1. **UI.HeaderInfo** is the title banner for the External Object.

**Super Star Holdings Inc** Featured in 1 Group(s)

The **UI.HeaderInfo** Annotation Term in this example consists of three **PropertyValue** Property attributes with the following attribute-values:

- **TypeName**: the label to use for the object type.
- **TypeNamePlural**: the label to use for the plural of the object type.
- **ImageUrl**: the URL to the icon representing the object type.
- **Title**: the mapping between the object title field and the external OData source. Note that the HeaderInfo Title is mapped to the "name" property of the OData Account resource.

```
<Annotations Target="SugarCRM.Opportunity">
  <Annotation Term="UI.HeaderInfo">
    <Record>
      <PropertyValue Property="TypeName" String="Opportunity" />
      <PropertyValue Property="TypeNamePlural" String="Opportunities" />
      <PropertyValue Property="ImageUrl" String="/images/exobj/
opportunity/opportunity_grey_48.png" />
      <PropertyValue Property="Title">
```

```

        <Record>
          <PropertyValue Property="Value" Path="name" />
        </Record>
      </PropertyValue>
    </Record>
  </Annotation>

```

2. **UI.Identification** contains the identifying details for the External Object.

Name :	Super Star Holdings Inc
Changed At :	2014-04-28 18:52:49
Account Type :	Customer
Industry :	Technology
Office Phone :	(910) 273-1181
Website :	www.infothe.org
Employee Responsible :	Sally Bronsen
Address :	345 Sugar Blvd. 52941 Sunnyvale USA

1. **Communication.Address**, which is included in UI.Identification in the following example, presents a mailing address block for use in other UI segments.

```

<Annotation Term="Communication.Address">
  <Record>
    <PropertyValue Property="street" Path="billing_address_street"/>
    <PropertyValue Property="locality" Path="billing_address_city"/>
    <PropertyValue Property="postalCode"
Path="billing_address_postalcode"/>
    <PropertyValue Property="country" Path="billing_address_country"/>
  </Record>
</Annotation>
<Annotation Term="UI.Identification">
  <Collection>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Name" />
      <PropertyValue Property="Value" Path="name" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Changed At" />
      <PropertyValue Property="Value" Path="date_modified" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Account Type" />
      <PropertyValue Property="Value" Path="account_type" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Industry" />
      <PropertyValue Property="Value" Path="industry" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Office Phone" />
      <PropertyValue Property="Value" Path="phone_office" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Website" />
      <PropertyValue Property="Value" Path="website" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Employee Responsible" />
      <PropertyValue Property="Value" Path="assigned_user_name" />
    </Record>
    <Record Type="UI.DataFieldForAnnotation">
      <PropertyValue Property="Label" String="Address" />
      <PropertyValue Property="Target"
AnnotationPath="@Communication.Address" />
    </Record>
  </Collection>

```

```
</Annotation>
```

The UI.Identification Annotation Term of the object details appears immediately below the HeaderInfo. It is described by the "UI.Identification" annotation term.

Each row of the Identification section should be described as a Record entry within an annotation Collection, as shown in the screen capture and the code example above. The row label and field bindings are described by the "Label" and "Value" properties in each Record.

The "Type" attribute of the Record also defines how each field is to be rendered in the Identification section. The supported values are shown in the code example above, which shows the options of UI.DataField and UI.DataFieldForAnnotation.

### 3. UI.Facets defines a set of tabs.



```
<Annotation Term="UI.Facets">
  <Collection>
    <Record Type="UI.ReferenceFacet">
      <PropertyValue Property="Label" String="Contacts" />
      <PropertyValue Property="Target" AnnotationPath="contacts/"
@UI.LineItem" />
    </Record>
    <Record Type="UI.ReferenceFacet">
      <PropertyValue Property="Label" String="Opportunities" />
      <PropertyValue Property="Target" AnnotationPath="opportunities/"
@UI.LineItem" />
    </Record>
    <Record Type="UI.ReferenceFacet">
      <PropertyValue Property="Label" String="Account Details" />
      <PropertyValue Property="Target" AnnotationPath="details/"
@UI.LineItem" />
    </Record>
  </Collection>
</Annotation>
```

### 4. UI.LineItem defines a row in a table of items.

Title	Changed At	Sales Stage	Closing Date	Exp. Sales Vol.	Chance of Success %
Super Star Holdings ...	2014-04-28 18:52:49	Qualification	2015-04-24	50000	60 %

```
<Annotation Term="UI.LineItem">
  <Collection>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Title"/>
      <PropertyValue Property="Value" Path="name"/>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Changed At"/>
      <PropertyValue Property="Value" Path="date_modified"/>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Sales Stage"/>
      <PropertyValue Property="Value" Path="sales_stage"/>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Closing Date"/>
      <PropertyValue Property="Value" Path="date_closed"/>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Exp. Sales Vol."/>
      <PropertyValue Property="Value" Path="amount"/>
    </Record>
    <Record Type="UI.DataField">
```



```

        <PropertyValue Property="Label" String="Chance of Success %"/>
        <PropertyValue Property="Value" Path="probability"/>
    </Record>
</Collection>
</Annotation>

```

Other [Term](#) attribute value examples are:

- **UI.FieldGroup**: enables the display of a set of fields in a facet that are not in tabular form.

Contacts		Opportunities		Account Details	
		<div> <div>Account Type : Customer</div> <div>Industry : Technology</div> <div>Search : <a href="#">Click here to search</a></div> </div>			
Name	E-Mail	Office Phone	Mobile Phone		
Carroll Moss	phone.sugar@example.us	(883) 302-1375	(024) 379-2999		

```

<Annotation Term="UI.FieldGroup" Qualifier="Industry">
  <Record>
    <PropertyValue Property="Label" String="Industry Details"/>
    <PropertyValue Property="Data">
      <Collection>
        <Record Type="UI.DataField">
          <PropertyValue Property="Label" String="Account Type" />
          <PropertyValue Property="Value" Path="account_type" />
        </Record>
        <Record Type="UI.DataField">
          <PropertyValue Property="Label" String="Industry" />
          <PropertyValue Property="Value" Path="industry" />
        </Record>
        <Record Type="UI.DataFieldWithUrl">
          <PropertyValue Property="Label" String="Search" />
          <PropertyValue Property="Value">
            <Apply Name="odata.concat">
              <String>search term: </String>
              <Path>industry</Path>
            </Apply>
          </PropertyValue>
          <PropertyValue Property="Url">
            <Apply Name="odata.concat">
              <String>http://www.google.com/custom?q=</String>
              <Path>industry</Path>
            </Apply>
          </PropertyValue>
        </Record>
      </Collection>
    </PropertyValue>
  </Record>
</Annotation>

```

- **Core.Description**:

```

<Annotation Term="UI.LineItem">
  <Collection>
    <Record Type="UI.DataField">
      <PropertyValue Property="Value" String="/[path_to_images]/[filename].png">
        <Annotation Term="UI.IsImageURL"/>
      </PropertyValue>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Product ID" />
      <PropertyValue Property="Value" Path="ProductID" />
    </Record>
  </Collection>
</Annotation>

```

```

    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Product" />
      <PropertyValue Property="Value" Path="ProductName">
        <Annotation Term="Core.Description"/>
      </PropertyValue>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Quantity / Unit" />
      <PropertyValue Property="Value" Path="QuantityPerUnit" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Unit Price" />
      <PropertyValue Property="Value" Path="UnitPrice" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Units In Stock" />
      <PropertyValue Property="Value" Path="UnitsInStock" />
    </Record>
  </Collection>
</Annotation>

```

- **UI.IsImageUrl:**

```

<Annotation Term="UI.LineItem">
  <Collection>
    <Record Type="UI.DataField">
      <PropertyValue Property="Value" String="/[path_to_images]/[filename].png">
        <Annotation Term="UI.IsImageUrl"/>
      </PropertyValue>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Product ID" />
      <PropertyValue Property="Value" Path="ProductID" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Product" />
      <PropertyValue Property="Value" Path="ProductName">
        <Annotation Term="Core.Description"/>
      </PropertyValue>
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Quantity / Unit" />
      <PropertyValue Property="Value" Path="QuantityPerUnit" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Unit Price" />
      <PropertyValue Property="Value" Path="UnitPrice" />
    </Record>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Units In Stock" />
      <PropertyValue Property="Value" Path="UnitsInStock" />
    </Record>
  </Collection>
</Annotation>

```

## Type attribute values

**Description:** **Type** is an attribute of the **Record** element. The various values of the **Type** attribute specify the UI widget type that is used to present the data mapped in this **Record** element. Each widget type requires certain pieces of data, as is shown in the following examples.

The valid values for the **Type** attribute are:

- **UI.CollectionFacet**

```
<Annotation Term="UI.Facets">
  <Collection>
    <Record Type="UI.CollectionFacet">
      <PropertyValue Property="Label" String="Supplier" />
      <PropertyValue Property="Facets">
        <Collection>
          <Record Type="UI.ReferenceFacet">
            <PropertyValue Property="Label" String="Contacts" />
            <PropertyValue Property="Target"
AnnotationPath="Supplier/@UI.Identification" />
          </Record>
          <Record Type="UI.ReferenceFacet">
            <PropertyValue Property="Label"
String="Opportunities" />
            <PropertyValue Property="Target"
AnnotationPath="Supplier/@vCard.Address" />
            <Annotation Term="UI.Map" />
          </Record>
        </Collection>
      </PropertyValue>
    </Record>
  </Collection>
</Annotation>
```

- **UI.DataField**

```
<Annotation Term="UI.Identification">
  <Collection>
    <Record Type="UI.DataField">
      <PropertyValue Property="Label" String="Category ID" />
      <PropertyValue Property="Value" Path="CategoryID" />
    </Record>
  </Collection>
</Annotation>
```

- **UI.DataFieldForAnnotation**

```
<Record Type="UI.DataFieldForAnnotation">
  <PropertyValue Property="Label" String="Supplier Adress" />
  <PropertyValue Property="Target" AnnotationPath="@vCard.Address" />
</Record>
```

- **UI.DataFieldWithNavigationPath**

```
<Record Type="UI.DataFieldWithNavigationPath">
  <PropertyValue Property="Label" String="Customer" />
  <PropertyValue Property="Value" Path="Customer/CompanyName" />
  <PropertyValue Property="Target" NavigationPropertyPath="Customer" />
</Record>
```

- **UI.DataFieldWithUrl**

```
<Record Type="UI.DataFieldWithUrl">
  <PropertyValue Property="Label" String="Homepage" />
  <PropertyValue Property="Value" Path="HomePage" />
  <PropertyValue Property="Url" Path="HomePage" />
</Record>
```

- **UI.ReferenceFacet**

```
<Record Type="UI.ReferenceFacet">
  <PropertyValue Property="Label" String="Overview" />
```

```
<PropertyValue Property="Target"
AnnotationPath="@UI.FieldGroup#Overview" />
</Record>
```

## Property attribute values

**Description:** **Property** is an attribute of the **PropertyValue** element. The various values of the **Property** attribute specify the type of data involved, whether that is a UI element or a particular piece of data from the external application's OData API.

The valid values for the **Property** attribute are:

- **ImageUrl**

```
<PropertyValue Property="ImageUrl" String="/images/exobj/account/
account_grey_48.png" />
```

- **Label**

```
<PropertyValue Property="Label" String="Product ID" />
```

- **Target**

```
<PropertyValue Property="Target" NavigationPropertyPath="Supplier" />
```

- **Title**

```
<PropertyValue Property="Title">
  <Record>
    <PropertyValue Property="Label" String="Category Name" />
    <PropertyValue Property="Value" Path="CategoryName" />
  </Record>
</PropertyValue>
```

- **TypeName**

```
<PropertyValue Property="TypeName" String="Supplier" />
```

- **TypeNamePlural**

```
<PropertyValue Property="TypeNamePlural" String="Suppliers" />
```

- **Value**

```
<PropertyValue Property="Value" Path="CompanyName" />
```

Once you have completed the development of your annotations XML file, you need to host it where it is publicly accessible from the internet (accessible to your SAP Jam instance). If you want, you can secure it with the same authentication credentials used to access the external application's OData service, although that is not necessary.

### i Note

**Next:**

Note that there are two sub-pages to this page that are important as prerequisites and debugging information in the development of your integration:

- [OData Assumptions \[page 41\]](#) describes the prerequisites for your external application's OData API, particularly for developing your annotations file.
- [External Application Error Messages \[page 49\]](#) describes the types of error messages that you may encounter at various stages of developing your external business records application integration, although again these error messages will largely be encountered when testing out your annotations file.

Once you have successfully developed your annotations file, you should [register the individual business records that you want to access in SAP Jam \[page 59\]](#).

## 2.1.7.1 OData Assumptions

### OData Versions

The Business Record Editor (BRE), the Business Record Viewer (BRV), and the Business Record Browser (BRB) all assume that the external application provides an OData version 2.0 API. For example, `$format=json` must be supported.

The above statement applies to what OData sources SAP Jam Collaboration supports. OData also defines the concept of "annotations" in V4. In that respect, the new BRV in SAP Jam will support OData v.2 data feeds, with a limited set of OData v.4 annotations (in a separate resource URL).

The expectation of the OData data source is that it will behave responsibly. That is, entities and properties can be added, but not removed. Properties can be deprecated (for example, made nullable). For the most part the API should be backwards compatible with previous functionality. The same approach is taken in the SAP Jam OData API.

### Timeouts

When making calls to the external application, we expect responses to occur in a reasonable period of time, although we do allow some leeway. Our official timeout on external calls is 5 seconds for opening a connection and 30 seconds on reading from that call. If a failure occurs we will try again, however, timing out on the same request twice will result in a three minute wait before the next retry is performed.

### Users Who Don't Have Access and External Users

Users who don't have access to the external application and external users should act similarly. That is they should be able to see the primary external object link in the left hand nav but once they click on it they

shouldn't see any data (but they should see the feed below). Additionally, the featured objects should be visible. Essentially, any live data should not be available, such as the lists of related objects (but the navigation could exist).

## Business Record Editor (BRE)

The BRE is responsible for importing external XML annotations and storing them locally within SAP Jam.

Current maximums:

- Identity: 20 fields, maximum.
- Lineltem: 10 fields, maximum.
- Quickview/Overview: 7 fields, maximum. Generated if missing as the first 7 items in the identity section.
- Facets: a total of 5 facets/tabs:
  - Detail Facet: 20 fields, maximum.
  - Collection Facet: 10 fields, maximum.

SAP Jam parses the following annotation terms for a given record type:

- `UI.HeaderInfo` (required)
  - The Title property is required. All other properties are optional.
- `UI.Identification` (required)
  - Must contain at least one parseable record that can be displayed in the BRV.
- `UI.Lineltem` (required)
  - Must contain at least one parseable record that can be displayed in the BRB.
- `UI.Facets`
  - SAP Jam tries to parse records of type `UI.ReferenceFacet` or `UI.CollectionFacet` within the `UI.Facets` collection.
  - The facet is expected to have a valid Label PropertyValue.
  - A `UI.ReferenceFacet` is expected to have a Target Property with an AnnotationPath.
    - SAP Jam accepts an AnnotationPath of the form "`@UI.FieldGroup#Qualifier`" or "`Navigation/@UI.LineItem`" or "`@Term/@UI.LineItem`".
    - The first case must refer to a `UI.FieldGroup` annotation belonging to the same entity type with the defined Qualifier.
    - The second case refers to a `UI.LineItem` annotation belonging to the navigation's entity type. Note that the navigation must be a collection.
    - The third case refers to an annotation term that references a collection from a linked annotation.
  - A `UI.CollectionFacet` is expected to contain at least one `UI.ReferenceFacet` that follows the previous rule.
    - SAP Jam will attempt to parse the first `UI.ReferenceFacet` sub-facet that follows this pattern and will display its content with the `UI.CollectionFacet`'s label.
    - All other subfacets in the `UI.CollectionFacet` will be ignored.

SAP Jam will try to parse Apply elements that use the "`odata.concat`" Function.

SAP Jam will try to parse `Measures.Unit` and `Measures.ISOCurrency` annotation terms.

## Linked Annotations

SAP Jam has very limited support for parsing linked annotations from other OData services.

SAP Jam tries to import linked annotation documents by analyzing the XPath:

```
//edmx:Reference[@Uri and edmx:IncludeAnnotations/@TargetNamespace]
```

SAP Jam tries to import the associated metadata resources by analyzing the XPath:

```
//edmx:Reference[@Uri and edmx:Include/@Namespace='TargetNamespace' and  
edmx:Include/@Alias]
```

The URI attributes for both the annotations and metadata must be fully qualified URIs. Relative URIs are not currently supported.

Any types that are referenced from the linked annotation must have a term defined under the main annotation schema with the following format:

```
<Term Name="TermIdentifier" Type="Collection(MetadataAlias.ForeignEntityType)"/>
```

When you want to show a collection of this foreign entity type in a facet, you must create a special annotation to define the URI of the collection following this format:

```
<Annotation Term="SchemaAlias.TermIdentifier">  
  <UrlRef>  
    <Apply Function="odata.fillUriTemplate">  
      <String>https://pattern...'{P0}'...pattern</String>  
      <LabeledElement Name="P0">  
        <Apply Function="odata.UriEncode">  
          <Path>NavigationPath</Path>  
        </Apply>  
      </LabeledElement>  
    </Apply>  
  </UrlRef>  
</Annotation>
```

Defining the URI using any other method is not supported.

The facet must reference this annotation by using:

```
<PropertyValue AnnotationPath="@SchemaAlias.TermIdentifier/@UI.LineItem"  
Property="Target"/>
```

The foreign entity type must have a `UI.LineItem` annotation defined in the linked annotation document.

## Business Record Viewer (BRV)

- The viewer assumes that the external annotation has been imported to SAP Jam. Any unsupported features are simply filtered out at import time (see above).
- The viewer only supports OData responses in JSON format, so all requests have the `$format=json` parameter appended.

- No `$expand` parameters are used, so if the annotations references any properties that aren't contained within the external object, for example, properties of navigations, the BRV will display an error message instead.
- Table content supports pagination of 20 items per page using `$top=20` and `$skip` to move between pages.
- If no `UILink` or `CRMUILink` is provided, there will be no permalink provided. In this case, the UI in the BRV will hide the link for "View in XXX".

#### i Note

We currently do not support "Precision" and "Scale" field in the property value. For all decimal types, we display the content as given without modification. For example, "1000.000" will be display as "1000.000".

#### i Note

For the SuccessFactors Learning (LMS) external application, we added a workaround to hide the "show more" button in the Business Record facet viewer. LMS does not currently support paging for facet collections."

## Business Record Browser (BRB)

The URL for the BRB is the URL from "External Type" on the object type with \$ to the # removed. For example, the following URL:

```
https://example.com/odata/v1/c4c.svc/$metadata#CorporateAccountCollection
```

would translate to:

```
https://example.com/odata/v1/c4c.svc/CorporateAccountCollection
```

#### i Note

The object type is defined in the "External Applications" section of admin for a company.

Assumptions on this endpoint and how the list gets shown:

- `$skip` and `$top` are supported on the above endpoint:
  - Potentially, it also needs to support `$filter` and `$orderby` (see below).
  - `$orderby` needs to support both `asc` (ascending) and `desc` (descending) options. This is assumed on sort fields.
- `lineItem` is defined in the annotation:
  - The first field of the line item is the "title"; it shows up first.
  - The second field is the second column.
  - All other fields show up under the title in that column, unless they are marked as sortable (see below).
- The BRB instantiates `ex_objs`, which are links back to the parent system. Since we only want to have one link back to the system, it must be unique. To achieve this, the result that is returned by the endpoint must have URLs that are consistent elsewhere in the application.



- There are two possible formats for even the most simple OData URL:

```
https://example.com/odata/v1/c4c.svc/
CorporateAccountCollection('00163E03A0701ED28B9DAF815866301D')
```

```
https://example.com/odata/v1/c4c.svc/
CorporateAccountCollection(Id='00163E03A0701ED28B9DAF815866301D')
```

Either one or the other must be used throughout the application.

- They should be made consistent with any calls made to APIs such as `POST /ExternalObjects`.
- A property called "Name" exists that we can use when instantiating an `ex_obj`.
- Non-Compounded data fields will have a navigation field which we can use as a path. For example, "&Name"
- Compounded data fields are specified by the params and pattern tags.
  - In the params list, there should be a navigation, which is used to build up the value and path.
  - There is no recursion in the params tag. If there was recursion, we will definitely will not have room on screen to display compounded data fields.
  - Paths are generated by walking the params and building the string using the navigation. For example, "&ExpectedValue/Content &ExpectedValue/Currency".
  - We do not support recursively generating compounded data field. It would not be feasible as we don't have sufficient screen real estate to render them.
- If a Date field uses the `Date (xx)` decoration, it will be rendered using the long form in the thing inspector with the following format: "%B %d, %Y, %I:%M:%S %p". The quick hover card will use the short form with the following format: "%b %d, %Y, %I:%M:%S %p"
- The date string is expressed in the time zone of the OData API user associated with the OAuth token.

#### To make the columns of your business record sortable:

1. Open the SAP Jam [Admin](#) [External Applications](#) section.
2. In the row for your application, click [Action](#) and select [Manage Record Types](#) from the drop-down menu.
3. In the row for the business record to which you want to add a sort property, click [Sort Fields](#).
4. Add the sort fields you want in the [Sort Fields](#) text box.  
Click [Show Fields Hint](#) to display a list of the properties available for your sort order. You can copy and paste these values into the [Sort Fields](#) text box. Ensure that each entry in the text box begins with an ampersand (&) and is separated from the next field by a comma.  
Ensure that your OData service supports "\$orderby=thePropertyYouJustAdded".
5. When you have set the sort order that you want, click [Update](#).

#### To apply filters to your business record:

1. Open the SAP Jam [Admin](#) [External Applications](#) section.
2. In the row for your application, click [Action](#) and select [Manage Record Types](#) from the drop-down menu.
3. In the row for the business record to which you want to add a filter, click [Filters](#).
4. In the filters catalog for the selected business record, click [New Filter](#).
5. Enter a [Name](#) for the filter, add properties in the [Filter](#) text box.  
Ensure that the properties you add are `$filterable` in your OData service.
6. Select the [Enabled](#) check box to make it immediately available, and click [Submit](#) to save the filter.

#### To make your business record searchable:

1. Open the SAP Jam [Admin](#) [External Applications](#) section.

2. In the row for your application, click [Action](#) and select [Manage Record Types](#) from the drop-down menu.
3. In the row for the business record to which you want to add search capabilities, click [Edit](#).
4. In the [Edit Record Type](#) dialog box, select the [Show Search](#) check box, and set the [Hint](#) (the example text that will appear in the empty search text box) and the [Property](#) that users searches will search against. If the property is a string, it must support `startswithfor "Edm.Int32", "Edm.Int64", "Edm.Int16", "Edm.Decimal", "Edm.Single", "Edm.Double"`.  
In the search operation, we just check for equality; for example: `startswith(AccountID, '100')` means that our string search starts from the start of a string (not an inner search).  
See `convert_search_string_to_odata_query` for more details.

## Related Business Record (Related External Objects)

When a business record has a one to many relationship with other business records through navigation paths, they are considered related. This is determined by parsing the metadata to find associated navigation paths to other business records. When relationships are found via the `navigation_path`, we add an entry in the `related_items` section of the view definition with the `type`, `navigation`, `entityType`, and `entityCollectionName`.

An example would look something like this:

```
"relatedItems": [
  {
    "type": "relatedItem",
    "navigation": "Opportunity",
    "entityType": "Opportunity",
    "entityCollectionName": "OpportunityCollection"
  },
  ..
]
```

When viewing a group page with a business record, we check if any of the related items specified in the `relatedItems` in the `view_definition` were registered previously with the client integration. If found, the related objects will be added to the related list, which is viewable by selecting the client integration on the left navigation of that group.

We currently assume the following:

- Related navigations do not use complex keys.
- Supported related objects are defined by a metadata `navigation_path` field.
- For performance purposes, we do not update related business records, including the business records that have no related record. Currently, admin users will have to go to the SAP Jam [Admin](#) [External Applications](#) section to manually re-import the `view_definition` to update and refresh the metadata and the annotations.
- Related objects that are defined by using the extra "term" field from the annotation are not currently supported.
- Related objects are also navigated to using only a single key `odata_link`.
- Navigations to related objects can have filters. For example, a related object with a navigation of Opportunity may have a navigation URI of:

```
"https://domain/EntitySet('00163E03A0701')/Opportunity?$format=json&
$orderby=ChangedOn+desc&$stop=20&$skip=20"
```

- When determining what sort property to use when navigating a related Object, the related object's view definition is used to look up the sort properties.

## Related Objects for CRM:

We added more complex support for CRM objects by adding support for linked metadata and complex navigation to the `relatedItem` section of the view definition. CRM allows related external objects to be of different object types than the main external object in the group. `linkedMetadataUri` was added to allow identification of the object type to navigate to. A more complex navigation, with interpolated strings, was also added.

For example:

```
{
  "type": "relatedItem",
  "navigation": {
    "type": "interpolatedString",
    "pattern": "https://example.com/odata/ESJI_SD_SRV;o=QI3CLNT503_T/
CustomerCollection(ObjectID='{0}')/Quotations",
    "params": [
      {
        "type": "oDataProperty",
        "oDataType": "Edm.String",
        "navigation": "CustomerNo"
      }
    ]
  },
  "entityType": "Quotation",
  "entityCollectionName": "QuotationCollection",
  "linkedMetadataUri": "https://example.com/odata/
ESJI_SD_SRV;o=QI3CLNT503_T/$metadata#QuotationCollection"
},
],
```

In this example, to work out what the navigation path is, retrieve the `CustomerNo`. Make a data call using the `linkedMetadataUri` to determine which external object type end point to call. Once the `CustomerNo` is retrieved from the data call, set the `ObjectID` as the `CustomerNo` to create the full URL. The full URL is then used to make a data call to get the related object.

For example, if `CustomerNo=5` is returned, then the URL we would call, constructed from this navigation pattern, would be:

```
"https://example.com/odata/ESJI_SD_SRV;o=QI3CLNT503_T/
CustomerCollection(ObjectID='5')/Quotations"
```

For CRM, anytime there is linked metadata in the facet, two calls must be made to retrieve the related information.

## Featured business records

To support featured business records, the remote URI is called to get the data to support more complex filtering. To retrieve data for the list of the featured business record from the external site, we build a URI with a

filter that selects all the business record ID keys. When building the filter, pay attention to single key vs multiple compounded keys. When the business record has a single key in the `odata_link`, use the property key to construct the filter with the ID. However, when a featured business record has multiple keys in the `odata_link`, it is not necessary to use the property key name with the filter.

Here are some examples of how the URI is created:

- Retrieve one featured business record with a single key format, with a property key of

"OpportunityUUID":

- `odata_link`:

```
https://example.com/OpportunityCollection('111111')
```

- `uri`:

```
https://example.com/OpportunityCollection(OpportunityUUID%20eq%20'111111')
```

- Retrieve multiple featured business record with a single key with a property key of "OpportunityUUID":

- featured business record 1 with `odata_link`:

```
https://example.com/OpportunityCollection('111111')
```

- featured business record 2 with `odata_link`:

```
https://example.com/OpportunityCollection('222222')
```

- `uri`:

```
https://example.com/OpportunityCollection(OpportunityUUID%20eq%20'111111'%20or%20OpportunityUUID%20eq%20'222222')
```

- Retrieve a featured business record with multiple keys, using the property keys already provided in the multi-key format, as follows:

- `odata_link`:

```
https://example.com/OpportunityCollection(ObjectID='111111',ObjectType="666666")
```

- `uri`:

```
https://example.com/OpportunityCollection(ObjectID%20eq%20'111111'%20and%20ObjectType%20eq%20'666666')
```

- `uri` with multiple objects with keys

```
https://example.com/OpportunityCollection(ObjectID%20eq%20'111111'%20and%20ObjectType%20eq%20'666666'%20or%20ObjectID%20eq%20'222222'%20and%20ObjectType%20eq%20'666666')
```

Using filters is the most efficient way to get a list of items from the external application, as it requires only a single call to get the specified results quickly. This does require that the request must be limited to the maximum number of characters allowed in the URI request. Also, the request must be limited in terms of the volume of data that is returned.

## Examples

Assuming that it is acceptable for the application to expose Opportunities, Accounts, and Service Tickets, the following endpoints will be called.

- **Getting Annotations:** This URI is from the `ex_obj_object` table.

```
https://example.com/odata/v1/c4c.svc/AnnotationCollection('aa')/Content/$value
```

- **Getting MetaData:** This URI is from the `ex_obj_object` table.

```
https://example.com/odata/v1/c4c.svc/$metadata#ServiceRequestCollection
```

- **Retrieving corporate account information from the Thing Inspector:**

```
https://example.com/odata/v1/c4c.svc/  
CorporateAccountCollection('00163E03A0701ED28B9DB004EAB8301D')/  
Opportunity?$format=json&$orderby=ChangedOn%20desc&$skip=0&$top=20
```

- **Getting a list of objects for Opportunities using filters:**

```
https://example.com/odata/v1/c4c.svc/OpportunityCollection?$filter=  
(ObjectID%20eq%20'00163E03A0701ED28BCEC7F4AA474109'%20or  
%20ObjectID%20eq%20'00163E03A0701ED28BCEC8A91C8DE109'%20or  
%20ObjectID%20eq%20'00163E03A0701ED28BCFF0A51EBEC395'%20or  
%20ObjectID%20eq%20'00163E03A0701ED28BCFF155FFEDA395'%20or  
%20ObjectID%20eq%20'00163E03A0701EE28BE049992A5DA8DB')&$format=json
```

## 2.1.7.2 External Application Error Messages

This section explains the SAP Jam Collaboration External Application error messages developed to help integrators deal with problems that are encountered when developing or troubleshooting external application integrations. These error messages are designed to assist integrations developers understand issues involving communications with the external applications, problems viewing external application metadata, and errors that are encountered when viewing OData Annotations in the Business Record Viewer (BRV), the Business Record Browser (BRB), the Business Record Editor (annotation importer), the Related Object page, the Feature Object page, the Quick Card pop-ups, and other view pages. The objective of SAP Jam External Application error messages is to give users, administrators, and integrators informative error messages that identify the source and nature of the issues encountered at any stage of the communications and data exchanges with external systems and the display of that data in the SAP Jam user and admin interfaces.

The goals of the SAP Jam External Application error messages subsystem are:

- To provide sufficient information to each level of end user to allow them to follow up with the issue, and to allow organizations to troubleshoot any issues that they encounter.
- To provide error messages that identify the source of the issue, whether that be the connection to the external application, problems with the external application's OData implementation, problems accessing or parsing the external application's OData metadata file, or problems encountered with the import process of the localized business record type annotation, or its formatting.
- To ensure that all areas of business record error reports or exceptions are caught and identified in sufficiently detailed text.

- To ensure that any external application error messages are forwarded and displayed to end users, SAP Jam company administrators, and integration developers.

#### **i Note**

Error messages should be available in each language selected when you add a business record. The error messages should match the language in use in that business record. However, if a language is used that is not enabled in the business record configuration, the language in which the error message is displayed will revert to the default language set for the business record.

#### **i Note**

Each of these error messages refers to an "Error Key". Users should quote these keys when reporting the encountered problem to their company administrators. External application integrators should quote these keys when reporting the encountered problem to their company administrators if they are unable to resolve the encountered problem using the provided error message information. Company administrators should quote these keys when and if they report the encountered problem or file a bug on the issue with SAP Jam Support.

The remainder of this section is a listing of the major types of SAP Jam External Application error messages, showing examples of:

- The display of error messages in the Annotations-filled UI objects
- The display of error messages in the SAP Jam group pages

## **SSL Error**

SSL Errors occur when an invalid SSL Certificate is provided to the external application.

## Import External Resources : Corporate Account (CRM)

Failure ⓘ Bulgarian

**SSL error occurred while connecting to server. You may need to manually add the server certificate to External Applications Trusted Certificate Authorities.**

**Error Detail**  
An internal network problem occurred with the the following error: "SSL\_connect returned=1 errno=0 state=SSLv3 read server certificate B: certificate verify failed"

**Error Key**  
c71fe1077b69444b8cdf

Failure ⓘ English

Could not import annotations. This business record type will not be created/updated.

---

### Import Summary

0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
2 Failures	Import failed.


You can reimport at any time if metadata or annotations change in the future.

Done

Importing Annotation

CRM / Corporate Account (CRM)

Showing: All ▾



**Error: 401**

SSL error occurred while connecting to server. You may need to manually add the server certificate to External Applications Trusted Certificate Authorities.

**Error Details**

An issue occurred. To troubleshoot, please provide the following error ticket number to your Jam administrator:  
72dd189b13855afe7ae9

BRV/BRB/Quick card

## OAuth Error

OAuth Errors occur when the external call failed to retrieve an OAuth token from the Gateway to make the OData call to the external system.

### Import External Resources : Account (C4C my307224)

Failure ⓘ

English

Import failed due to the following errors:

Could not retrieve OAuth token from Gateway.

**Error Detail**  
An internal network problem occurred with the the following error: 'Member: 1 is not from a provisioned SFSF company so unable to request assertion'  
**Error Key**  
71546bd8f465f1292cd3

Could not import annotations. This business record type will not be created/updated.

Import Summary

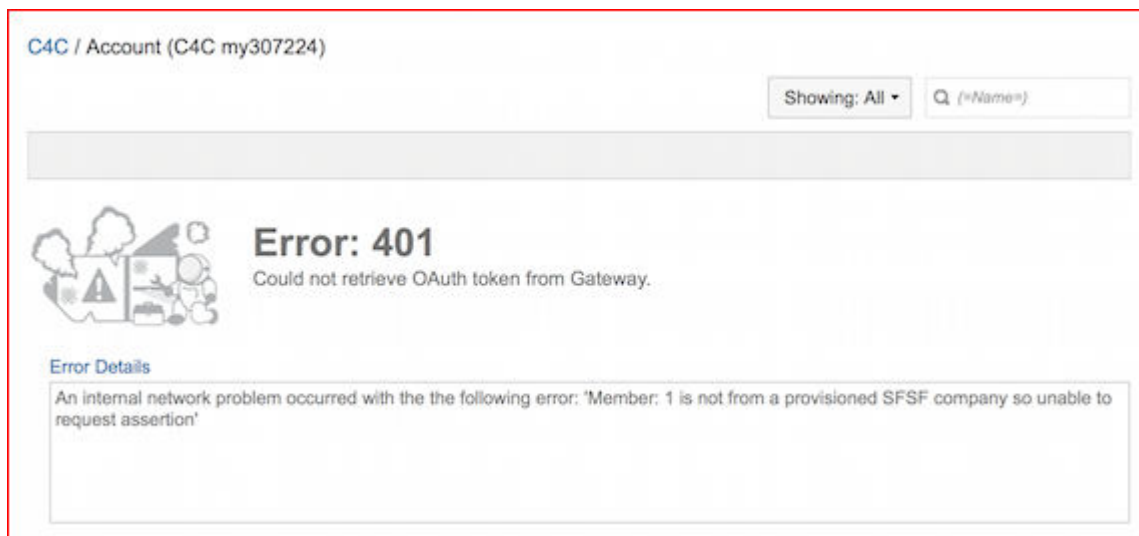
0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
1 Failure	Import failed.

You can reimport at any time if metadata or annotations change in the future.

Done

Importing Annotation





BRV/BRB/Quick card

## Time Out Error

Time Out Errors occur when the external application is taking too long to respond.

## Import External Resources : Account (C4C my307224)

Failure ⓘ English

**Import failed due to the following errors:**

**The 'C4C' timed out and is unavailable. Please try again later or contact your 'C4C' administrator for assistance.**

**Error Detail**  
An internal network problem occurred with the the following error: 'Net::ReadTimeout'

**Error Key**  
22f36961be6e761ed1d3

Could not import annotations. This business record type will not be created/updated.

---

### Import Summary

0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
1 Failure	Import failed.


You can reimport at any time if metadata or annotations change in the future.

Done

Importing Annotation

C4C / Account (C4C my307224)

Showing: All ▾ Q (=Name=)



**Error: 500**  
The 'C4C' timed out and is unavailable. Please try again later or contact your 'C4C' administrator for assistance.

BRV/BRB/Quick card

## Socket/Protocol Error

Socket/Protocol Errors occur when the connection fails to retrieve a socket or an invalid protocol is used.

## Import External Resources : Corporate Account (CRM)

Failure ⓘ Bulgarian

**Failed to connect with the external application due to a network socket error.**

**Error Detail**  
An internal network problem occurred with the the following error: 'getaddrinfo: nodename nor servname provided, or not known'

**Error Key**  
2cfc96d30fe28a753c94

Failure ⓘ English

Could not import annotations. This business record type will not be created/updated.

---

### Import Summary

0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
2 Failures	Import failed.


You can reimport at any time if metadata or annotations change in the future.

**Done**

Importing Annotation

CRM / Corporate Account (CRM)

Showing: All ▾

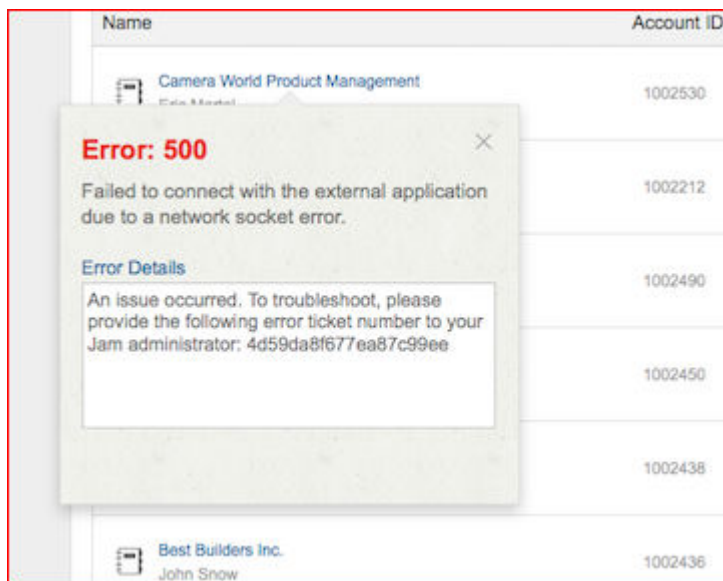


**Error: 500**  
Failed to connect with the external application due to a network socket error.

**Error Details**

An issue occurred. To troubleshoot, please provide the following error ticket number to your Jam administrator:  
386491f8be891827ae15

BRV/BRB/Quick card



BRV/BRB/Quick card (2)

## Standard Error

Standard Errors occur when there is some run-time crash or failure caused by code relating to invalid functions, variables, arguments, or IO, for example.

## Import External Resources : Account (C4C my307224)

Failure ⓘ English

**A system error occurred. Please contact your SAP Jam company or support administrator.**

**Error Detail**  
The external application returned the following error:'401', message:'uninitialized constant BusinessRecordLoader::SAPGatewayLoader::Somebadcode2'.

**Error Key**  
4339adf86c8fcf77e717

Could not import annotations. This business record type will not be created/updated.

---

### Import Summary

0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
1 Failure	Import failed.


You can reimport at any time if metadata or annotations change in the future.

Done

Importing Annotation

C4C / Account (C4C my307224)

Showing: All ▾ Q (=Name=)



## Error: 500

A system error occurred. Please contact your SAP Jam company or support administrator.

**Error Details**

An issue occurred. To troubleshoot, please provide the following error ticket number to your Jam administrator:  
bc72671d7b8a3ee99ca8

BRV/BRB/Quick card

## Generic Error

Generic Errors occur when there is an unknown run-time error or exception that was not caught by any of the above handlers. A ticket is provided that can help investigate the underlying problem if reported.

### Import External Resources : Account (C4C my307224)

Failure ⓘ

English

Import failed due to the following errors:

A system error occurred. Please contact your Jam administrator.

**Error Detail**  
An internal network problem occurred with the the following error: 'NoMemoryError'

**Error Key**  
6031bc858cdb3611a662

Could not import annotations. This business record type will not be created/updated.

---

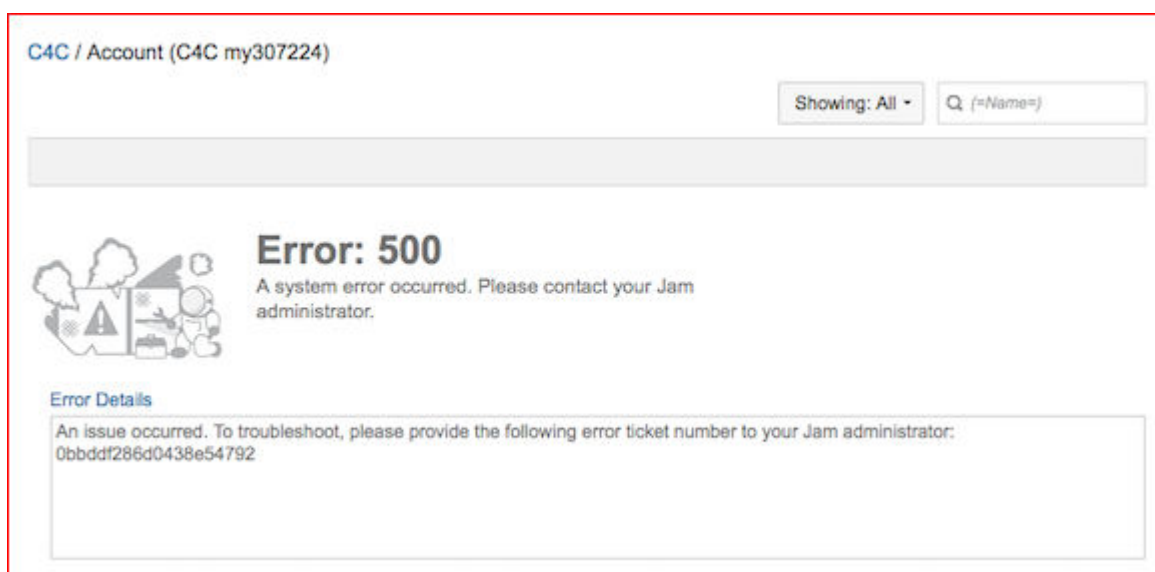
### Import Summary

0 Success	Import succeeded.
0 Warning	Import succeeded with warnings.
1 Failure	Import failed.

You can reimport at any time if metadata or annotations change in the future.

Done

Importing Annotation



BRV/BRB/Quick card

## 2.1.8 Register the business records in SAP Jam Collaboration

Once you have completed the required configuration to access your external application from SAP Jam Collaboration, you need to further configure it by adding the business objects (the annotations files), which are the specific sections of content that you want to access.

**To register business records in SAP Jam:**

1. In SAP Jam, as a company administrator, click on the cog icon at the top of the page and select [Admin](#) from the context menu.  
The SAP Jam [Admin](#) console displays.
2. In the [Integrations](#) [External Applications](#) section, click on [Action](#) beside the OData-based application for which you want to add objects and select [Manage Record Types](#) from the context menu.  
The [Manage <company\\_name> Record Types](#) screen displays for that integrated application.
3. Click [Add Record Type](#).  
The [Add Record Type](#) dialog box is displayed.

## Add Record Type

Name

External Type

Annotation URL

Annotation Languages   
English (Primary) X

Must provide valid type and annotation URLs before importing external resources. The record type cannot be created/updated without a successful import.

<input checked="" type="checkbox"/> Can be featured or unfeatured in groups <input type="button" value="i"/>	<input checked="" type="checkbox"/> Filter Feed <input type="button" value="i"/>
<input checked="" type="checkbox"/> Can be primary object in top-level group <input type="button" value="i"/>	<input checked="" type="checkbox"/> Feed History <input type="button" value="i"/>
<input checked="" type="checkbox"/> Can be primary object in subgroup <input type="button" value="i"/>	<input type="checkbox"/> Show Search
<input checked="" type="checkbox"/> Can be mentioned in feed posts <input type="button" value="i"/>	Hint: <input type="text"/>
Primary (Search) Property: <input type="text"/>	Property: <input type="text"/>
Secondary (Display) Property: <input type="text"/>	

### Add Record Type page

4. In the [Name](#) field, enter a meaningful name for the business record that you want to add.
5. Enter the [External Type](#) for the business record that you want to add.  
The [External Type](#) must be the URL to OData metadata EntityType of the business record type that you are adding.
6. Enter the [Annotation URL](#) for the type of object that you want to add.  
The [Annotation URL](#) must be the URL to the OData annotations file that you developed in the preceding procedure, [Develop an OData annotations file to display business records \[page 23\]](#). The annotations file describes how the incoming data will be laid out in a graphical element of the UI.  
If you click on the [Annotation URL](#) button, a drop-down menu displays that offers an alternate option of adding [Inline XML](#). This option provides a text box into which you, as an SAP Jam administrator, can paste an XML annotations file that provides an alternate mapping of the business record data into the business record UI elements, such as not showing the data for a particular column of information in a table of business records information. Note, however, that if you use the [Inline XML](#) option, you are restricted to displaying the external business records in English only.
7. Set the [Annotation Languages](#):
  - To add a language, click [Add Language](#) and select the language that you want from the drop-down menu.
  - To remove a language, click the "X" in that language's button.



- To set a language as the "Primary" language, remove all other languages, or remove all languages and add the language that you want to be the primary language first.

#### Language options

8. Select [Import External Resources](#) to import the data using the URLs listed above.  
This allows you to validate the URLs shown in the [External Type](#) and the [Annotations URL](#) fields. SAP Jam Collaboration will attempt to import the indicated resources and it will display a message indicating the success or failure of the import, as well as information on the cause of a failure if one occurs and the cause can be determined.
9. Select [Can be featured or unfeatured in groups](#) to enable this option.
10. Select [Can be primary object in top-level group](#) to enable this option.
11. Select [Can be primary object in subgroup](#) to enable this option.
12. Select [Filter Feed](#) to enable users to add or remove feed entries of this object type.
13. Select [Feed History](#) to see the feed history independently of follows, which means:
  - If disabled, the user will only see the subset of feed items that were routed to their home feed; that is, they will only see items which had a distribution list that included them, or items in which the user was at-mentioned.
  - If enabled, and if authorization is successful, the user will see the full feed for the object, including items that would otherwise not be visible to them.
14. Select [Show Search](#) to enable keyword searching in the business data browser.  
When rendering a list of External Objects, SAP Jam may also show a "search" box. If this option is selected, you can also set:
  - In the [Hint](#) field, you can enter a display string that tells the user what field the search is performed on.
  - In the [Property](#) field, you can enter the actual OData property name that the search is performed on.
15. Select [Can be mentioned in feed posts](#) to enable this option. If this option is selected, you can also set:
  - [Primary \(Search\) Property](#): enter the OData property name that an at-mention look-up will search for.
  - [Secondary \(Display\) Property](#): enter the OData property name that an at-mention look-up will display.
16. Once you have configured the record to your satisfaction, click [Save](#).  
You are returned to the [<External\\_App\\_Name> Record Types](#) catalog, and the record type that you just added will be listed in the table.

#### Note

Next:

You can now, optionally, [add an External Application Object Filter \[page 62\]](#) to narrow the range of data shown in a particular external application data object.

Additionally, you can now, optionally, [add an External Application Object Sort Order \[page 63\]](#) to set the order in which the external data is arranged in a particular external application data object.

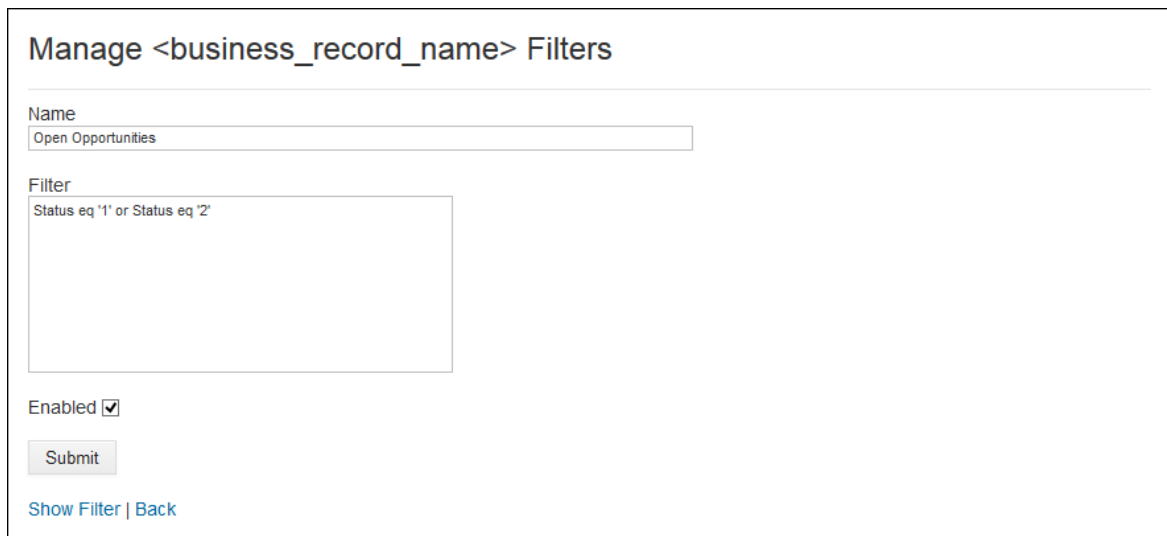
If you do not want or need to add Filters or Sort Fields to your External Application Objects, you can proceed to [Configuring access to business records in SAP Jam Collaboration \[page 64\]](#) or [Manage "external objects" \(business records\) using the SAP Jam Collaboration API \[page 65\]](#).

## 2.1.9 Configure a business record filter

Many External Application Objects come with predefined filters that narrow the range of data that is displayed in a UI object.

**To add an external object filter:**

1. As an SAP Jam Collaboration administrator, in the [Jam Collaboration Admin](#) [External Applications](#) page, with the list of external applications displayed, click on [Action](#) beside the external application to which you want to add an external object filter, and select [Manage Objects](#) from the context menu. The [Manage <External\\_Application> Objects](#) page is displayed.
2. Click [Filters](#) beside the object type to which you want to add a filter. The [Manage <object\\_type> \(<External\\_Application>\) Filters](#) page is displayed.
3. Click [New Filter](#). The [Manage <object\\_type> \(<External\\_Application>\) Filters](#) add a new filter page is displayed.



**Manage <business\_record\_name> Filters**

Name  
Open Opportunities

Filter  
Status eq '1' or Status eq '2'

Enabled ☒

Submit

[Show Filter](#) | [Back](#)

The add a new filter (Manage <business\_record\_name> Filters) page

4. In the [Name](#) field, enter a meaningful name for the filter that you want to add.
5. In the [Filter](#) field, enter the filter expression that you want to use.
6. Optionally, you can set the filter to [Enabled](#), which will make it immediately available.
7. When you are done, click [Submit](#).  
You are returned to the [Manage <object\\_type> \(<External\\_Application>\) Filters](#) panel.

8. In the *Manage <object\_type> (<External\_Application>) Filters* panel, you can do the following:
  - Click *true* (or *false*) in the *Enabled* column to toggle the filter's enabled status.
  - Click *Show Filter* to view the text of the filter.
  - Click *Delete Filter* to remove the filter, including from future use.
  - Click *Edit Filter* or to modify an existing filter.

## i Note

### Next:

You can now, optionally, [add an External Application Object Sort Order \[page 63\]](#) to set the order in which the external data is arranged in a particular external application data object.

If you do not want or need to add Sort Fields to your External Application Objects, you can proceed to [Configuring access to business records in SAP Jam Collaboration \[page 64\]](#) or [Manage "external objects" \(business records\) using the SAP Jam Collaboration API \[page 65\]](#).

## 2.1.10 Configure a business record sort order

Each business object type contains a set of attributes that can be displayed and sorted in SAP Jam Collaboration.

### To add an external object sort field:

1. As an SAP Jam administrator, in the [Jam Collaboration Admin > External Applications](#) page, with the list of external applications displayed, click on *Action* beside the external application to which you want to set sort fields, and select *Manage Objects* from the context menu.  
The *Manage <External\_Application> Objects* page is displayed.
2. Click *Sort Fields* beside the object type to which you want to set the sort fields.  
The *Edit Object* dialog box is displayed.

### Edit Record Type

Sort Fields · [Reset](#)

&ChanceOfSuccess,&ExpectedClosingDate,&ExpectedSalesVol,&LastModified,&StatusText

[Hide Fields Hint](#)

Available Sort Fields

&CalcChanceOfSuccess  
 &ChanceOfSuccess  
 &CreditStatus  
 &CreditStatusText  
 &Currency

Update

Cancel

Add a business record sort order

3. Click [Show Fields Hint](#) to view the list of available fields.  
A list of the available fields is displayed.
4. Copy and past the fields from the list of those available into the [Sort Fields](#) text box in the order in which you want them displayed.
5. When the sort fields are listed in the order in which you want the business records sorted, click [Update](#).  
You are returned to the [Manage <External\\_Application> Objects](#) page.

### i Note

**Next:** You can now proceed to [Configuring access to business records in SAP Jam Collaboration \[page 64\]](#) or [Manage "external objects" \(business records\) using the SAP Jam Collaboration API \[page 65\]](#).

## 2.1.11 Configuring access to business records in SAP Jam Collaboration

Once you have properly registered your external application, created any required annotations files, and registered your business records, your external data will be available for use in SAP Jam Collaboration. OData-based applications are listed in the [Business Records](#) page. You can also configure SAP Jam to access your organization's external content from other locations within SAP Jam.

- **To create a group with a business record as its "Primary Object":**
  1. In your SAP Jam [Home](#) page, click [Business Records](#), and click the name of the type of business record.  
A list of business records of that type is displayed.
  2. In the list of business records of that type, hover over the name of the particular business record that you want to set as the primary object of a new group.  
An information and actions dialog box is displayed for that business record
  3. Click [Create Group](#) at the bottom of that dialog box.  
A [Create a Group](#) dialog box is displayed.
  4. Set the options that you want for the group, including:
    - Select the group template.
    - Enter a group name
    - Optionally, enter a description of the group.
    - Click to select the [Group Permissions](#).
    - Click [options](#) beside the group permissions type for additional settings.
    - Ensure that the [Activate this group now](#) checkbox is selected if you want to allow immediate use of the group.
    - Click [Create](#) to create the group.
- **To "feature" a business record in a group:**
  1. In your SAP Jam [Home](#) page, click [Business Records](#), and click the name of the type of business record.  
A list of business records of that type is displayed.
  2. In the list of business records of that type, hover over the name of the particular business record that you want to set as the primary object of a new group.  
An information and actions dialog box is displayed for that business record.
  3. Click [Feature in...](#) at the bottom of that dialog box.  
A [Feature in another group](#) dialog box is displayed.
  4. Enter the name of the group in which you want to feature the selected business record and click [OK](#).

The selected business record is "featured" in the specified group.

- **To have a business record appear as being "related" to a "Primary" or "Featured" business record:**  
This is done automatically, but it does require that the UI.LineItem Term has been developed.

Your Document Repository access configuration is now complete. Users can access the newly integrated business records from any of the above listed SAP Jam locations that you configured.

#### i Note

**Next:** You may want to familiarize yourself with how external applications can be manipulated using the SAP Jam API, which is introduced in the following page, [Manage "external objects" \(business records\) using the SAP Jam Collaboration API \[page 65\]](#).

## 2.1.12 Manage "external objects" (business records) using the SAP Jam Collaboration API

Up to this point in the guide, we have only discussed exposing external objects in the SAP Jam Collaboration interface, via OData services and annotations. However, External Objects can also be created in SAP Jam, and added to a group as either the group's primary or related object. Feed entries can also be posted to External Object feeds, and retrieved by an external application whose users have access to the feed's external object. This section will discuss how to enable these features in an external application.

### Creating an External Object

External Objects are created in SAP Jam through an HTTP POST operation to the OData collection given by the URL: `/api/v1/OData/ExternalObjects`. For the API Reference page for this API call, see [\[POST\] / ExternalObjects](#).

The format of the POST request is provided by the OData v2 standard. SAP Jam accepts both JSON and XML requests. The parameters for the POST body are:

Property	Type	Use	Description	Nullable	Updatable
Exid	String	Optional	The external ID of the ExternalObject. This includes the URL that contains the "ObjectId" number.	False	False
Name	String	Mandatory	The name of the ExternalObject, which is used in feed postings.	False	False
Summary	String	Optional	A description of the ExternalObject.	False	False
Permalink	String	Optional	A URL to the ExternalObject's display in the external system.	False	False

Property	Type	Use	Description	Nullable	Updatable
<b>ODataAnnotations</b>	String	Optional	The OData annotations for the ExternalObject. This is a required property for the SAP Jam Extensions program, which includes work patterns, object rendering inside SAP Jam, and dynamic rights checking. These values must match the <a href="#">External Type</a> entry for the ExternalObject as set in the <a href="#">SAP Jam Collaboration Admin console &gt; External Applications &gt; Manage Objects &gt; Add Object</a> dialog box.	False	False
<b>ODataMetadata</b>	String	Optional	The URL to the external system's OData \$metadata resource for the ExternalObject. This is required to create a FeaturedExternalObject link and for the SAP Jam Extensions program, which includes work patterns, object rendering inside SAP Jam, and dynamic rights checking. These values must match the entries set in the SAP Jam <b>External Application</b> administration window.	False	False
<b>ODataLink</b>	String	Mandatory	The OData URL used to fetch the ExternalObject's data. This is a required property for the SAP Jam Extensions program, which includes work patterns, object rendering inside SAP Jam, and dynamic rights checking. These values must match the entries set in the SAP Jam <b>External Application</b> administration window.	False	False
<b>ObjectType</b>	String	Mandatory	The OData URL for the type of the ExternalObject in the external system. This is a required property for specifying the object that must match the <a href="#">External Type</a> entry for ExternalObject as set in the <a href="#">SAP Jam Collaboration Admin console &gt; External Applications &gt; Manage Objects &gt; Add Object</a> dialog box. It must be unique per client application.	False	False

The OData response body for the POST request will include the SAP Jam ID for the external object. This ID should be used in subsequent External Object calls to the SAP Jam OData API.

## Creating a Group with an External Object

Once the External Object is created in SAP Jam, it can be linked to a SAP Jam group. The object can either be linked as the group's "Primary" object, or it can be added into the set of "Featured" objects in the group.

Since a group can only have a single primary object, and since the object cannot be changed once the group is created, linking an object to a group as that group's "Primary" object can only be done at group creation time. Groups are created through an HTTP POST operation to the OData collection given by the URL: `/api/v1/OData/Groups`. For the API Reference page for this API call, see [\[POST\] /Groups](#).

The details of creating a Group with the SAP Jam OData API can be found in the API Guide. To include a link to an External Object as the group's primary object, the POST body should include the full URI to the SAP Jam representation of the External Object, following the OData standard for posting links. For example, the POST body for creating a group with a primary external object in JSON format should look like this:

```
{
  "Name": "Example Group",
  "Description": "This group has a primary external object",
  "GroupType": "private_internal",
  "PrimaryExternalObject": { "__metadata": { "uri": "https://<jam_uri>/api/v1/OData/ExternalObjects('<id>') " } }
}
```

## Featuring an External Object in a Group

External Objects can also be “Featured” into an existing group. To feature an object into an existing group, POST the object's URI to the URL, using the OData “InsertLink Request” format:








```
/api/v1/OData/Groups('group_id')/$links/FeaturedExternalObjects
```

In JSON, for example, the POST body would simply be:

```
{
  uri: "https://<jam_uri>/api/v1/OData/ExternalObjects('<id>') "
}
```

## SAP Jam OData API calls related to External Objects

The follow links point to the individual API calls and the sections in the SAP Jam OData API Reference that relate to external objects:

- The [External](#)  section in the API Reference.
- The [\[POST\] /Groups](#)  API call creates groups, including creating a group with a primary external object.
- The [\[GET\] /Groups\('{id}'\)/PrimaryExternalObject](#)  API call retrieves the information about the specified primary external object within the specified group.
- The [\[POST\] /Groups\('{id}'\)/\\$links/FeaturedExternalObjects](#)  API call adds an external object to the group as a featured external object.
- The [\[GET\] /Groups\('{id}'\)/FeaturedExternalObjects](#)  API call retrieves the information about the featured external objects within the specified group.
- The [\[DELETE\] /Groups\('{id}'\)/\\$links/FeaturedExternalObjects\('{id1}'\)](#)  API call removes an external object from the group as a featured external object.
- The [OData Activity API Calls](#)  are used to communicate external object Activity messages.

## 3 Branding

### 3.1 Custom Headers

Custom Header tags

Tag	Description
<code>&lt;jam-search&gt;&lt;/jam-search&gt;</code>	Provides filtered search functionality. A dropdown filter beside the search box helps users refine their search.
<code>&lt;jam-profile&gt;&lt;/jam-profile&gt;</code>	Displays the current user's profile avatar, access to user profile page, let's user set online presence and enable browser notifications.
<code>&lt;jam-tasks&gt;&lt;/jam-tasks&gt;</code>	Represents the clipboard icon used to access a user's tasks. Number of assigned or overdue tasks for current user.
<code>&lt;jam-notification&gt;&lt;/jam-notification&gt;</code>	Represents the bell icon used to access a notifications list. Shows the number of unread notifications for current user.
<code>&lt;jam-messaging&gt;&lt;/jam-messaging&gt;</code>	Represents the conversation icon used to access messages. Shows the number of unread messages for current user.
<code>&lt;jam-settings&gt;&lt;/jam-settings&gt;</code>	Represents the cog icon used to access account settings, the Admin console (only for company and area admins), and to log out.
<code>&lt;jam-help&gt;&lt;/jam-help&gt;</code>	Represents the Help icon used to access help pages and a getting started tour.
<code>&lt;jam-string&gt;&lt;/jam-string&gt;</code>	For content that has already been translated in one or more languages for the custom header for company, area, and external home pages.
<code>&lt;jam-instance-name&gt;&lt;/jam-instance-name&gt;</code>	Displays the product instance name.
<code>&lt;jam-company-logo&gt;&lt;/jam-company-logo&gt;</code>	Displays the company logo image.

Site Header tags

Tag	Description
<code>&lt;jam-navigation&gt;&lt;/jam-navigation&gt;</code>	Global navigation bar. This element accepts nested HTML and navigation elements. The tags below must be used inside <code>&lt;jam navigation&gt;</code>
<code>&lt;jam-home&gt;&lt;/jam-home&gt;</code>	Link to the company home page.
<code>&lt;jam-groups&gt;&lt;/jam-groups&gt;</code>	Opens the Group dropdown menu.



Tag	Description
<code>&lt;jam-knowledge-base&gt;&lt;/jam-knowledge-base&gt;</code>	Link to Knowledge Base articles page.
<code>&lt;jam-integrations&gt;&lt;/jam-integrations&gt;</code>	Link to Integrations page.
<code>&lt;jam-recommendations&gt;&lt;/jam-recommendations&gt;</code>	Link to the Recommendations page.
<code>&lt;jam-bookmarks&gt;&lt;/jam-bookmarks&gt;</code>	Link to the Bookmarks page.
<code>&lt;jam-calendar&gt;&lt;/jam-calendar&gt;</code>	Link to the Events page.
<code>&lt;jam-nav-item&gt;&lt;/jam-nav-item&gt;</code>	Add custom links to the navigation bar.

## 4 OpenSocial Gadgets

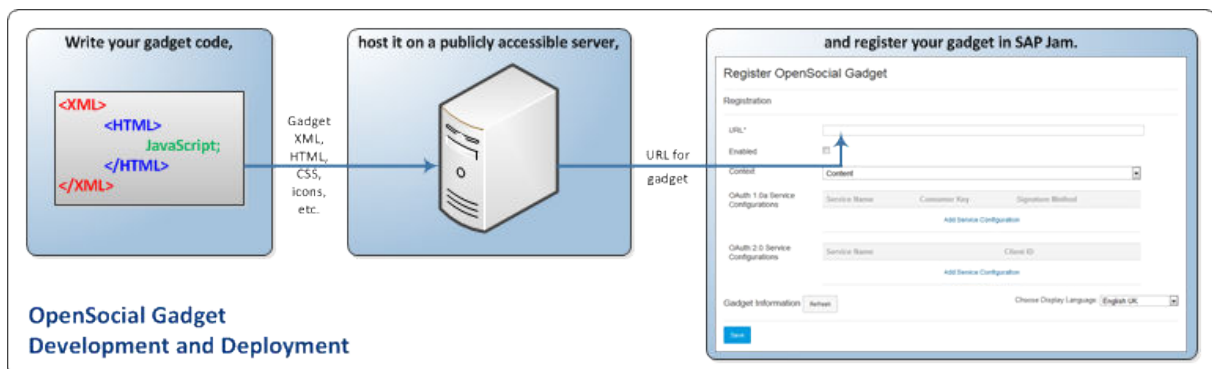
### 4.1 Develop SAP Jam OpenSocial gadgets

#### What is an OpenSocial gadget?

OpenSocial gadgets are web apps that extend the capabilities of SAP Jam Collaboration to interact with any software or service. Simply build or use an existing web app, wrap it in an XML container, and connect it to SAP Jam Collaboration with our OpenSocial and OData APIs.

Creating an OpenSocial gadget for use in SAP Jam involves both development and deployment tasks. This documentation introduces SAP Jam's support for OpenSocial gadgets, provides a quick tutorial to get you started, and includes reference pages for each SAP Jam-specific development and deployment step that is required to get your OpenSocial gadget up and running in your SAP Jam service.

In general, the steps required to develop and deploy your OpenSocial gadget on SAP Jam are:



#### Develop your SAP Jam OpenSocial gadget

To create an OpenSocial gadget for SAP Jam, you must develop the following three components:

- **XML:** An XML file, or wrapper, that provides metadata about the HTML, CSS, and JavaScript of the OpenSocial gadget application. This includes such information as how to process and render the gadget, and what features to include.
- **HTML:** Some simple HTML that is used to display the static content of your gadget in an SAP Jam page.
- **JavaScript:** The JavaScript that provides the business logic and dynamic behavior for your gadget. The JavaScript code must be based on SAP Jam's implementation of the OpenSocial APIs. The SAP Jam OpenSocial API is a Shindig implementation of the OpenSocial API that allows developers to create OpenSocial gadgets that they can register with SAP Jam. SAP Jam implements the OpenSocial

Core Gadget Specification, version 2.5. It includes API references for the supported Gadgets API, OpenSocial API, and Apache Wave API calls.

Note that, by default, when the data within a gadget is updated on one client, the updates are not made to other clients that display the same gadget within SAP Jam until those clients are refreshed. To allow updates to a gadget's data to be displayed immediately to other clients, you must use a [Wave Participants function](#) [page 184].

## Deploy your SAP Jam OpenSocial gadget

To deploy your OpenSocial gadget for SAP Jam, you must:

1. Host your gadget—the XML, HTML, and JavaScript, as well as any CSS and images—on a publicly accessible server.  
Gadgets cannot be directly uploaded to SAP Jam. Therefore, they must be uploaded to, and hosted on, an external server, not hosted within your private or corporate LAN. That is, you must ensure that the gadget's XML, HTML, CSS, and JavaScript files are visible on the public internet so that SAP Jam can access them.
2. Register the URL and other access particulars for your OpenSocial gadget by selecting the ► [Cog Icon](#) ► [Admin](#) ► [Integrations](#) ► [OpenSocial Gadgets](#) ► [Add Gadget](#) ► to get to the SAP Jam [Register OpenSocial Gadget](#) page. The URL must point to the XML file for your gadget.

When you register your SAP Jam OpenSocial Gadget-compliant application, you can configure it to have one of four types of [Contexts](#):

- **Content:** Content gadgets can be added to a group's Content section by anyone in the company with the appropriate privileges. They appear as an addable option in the Content area of SAP Jam Collaboration groups.
- **Profile:** Profile gadgets appear on the profile of each member of the company automatically. They appear in the ► [Product Setup](#) ► [Custom Profile](#) ► page, where you can configure where they will appear in your users' profile pages.
- **Group:** Group gadgets can be added to a group by anyone in the company with the appropriate privileges. They appear as a selectable list item in the left panel area of a SAP Jam Collaboration group.
  - Select ► [Group Admin](#) ► [Edit Group](#) ► [Setup](#) ► [Select Primary Group Extension](#) ► to add a group gadget to a group.
  - **Note:** Only one group gadget can be added to a group and it cannot be removed from the group it has been added to.
- **Statusbar:** Statusbar gadgets can be added by anyone in the company with the appropriate privileges. They appear on the bottom of every screen as an interactive button that can show alerts, badges and expand into a collapsible window.

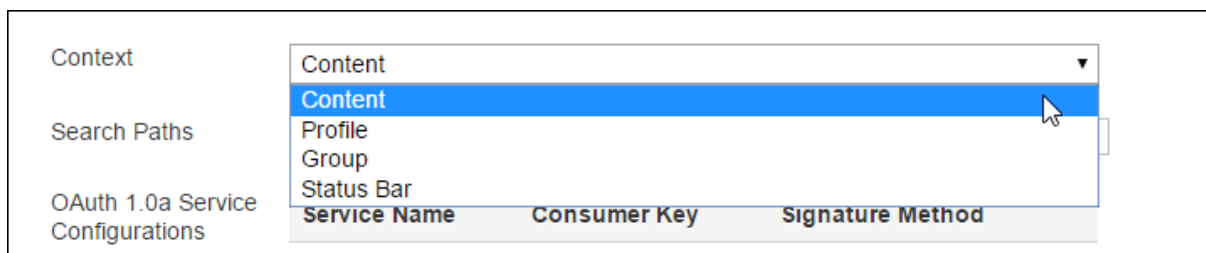
Context	<div>Content ▼</div> <div>Content</div> <div>Profile</div> <div>Group</div> <div>Status Bar</div>			
Search Paths				
OAuth 1.0a Service Configurations	<table><thead><tr><th>Service Name</th><th>Consumer Key</th><th>Signature Method</th></tr></thead><tbody></tbody></table>	Service Name	Consumer Key	Signature Method
Service Name	Consumer Key	Signature Method		

Whether your gadget is configured as a Content, Profile, Group, or Statusbar gadget these gadget contexts have some significant implications for where it is displayed, who can add it and delete it, how it behaves when it is deleted, and what data is displayed in it. The last of these implications is important to consider during the development of your gadget. These differences are discussed in the following topic:

## 4.1.1 Gadget Contexts

This page describes the differences between the behavior and handling of Content, Profile, Group, and Statusbar gadget contexts.

When an SAP Jam Collaboration company administrator registers an OpenSocial gadget in the SAP Jam Admin console, they must set the *Context* to be either "Content", "Profile", "Group", or "Statusbar".



Context	Content
Search Paths	Profile
OAuth 1.0a Service Configurations	Group
	Status Bar
	Service Name
	Consumer Key
	Signature Method

These options differ in the following ways:

- **Render location:**
  - A *Content* gadget, like any other group content item, can be added in a group's content section, and is rendered in its own single item view.
  - A *Profile* gadget renders on each user's profile page, below their expertise tile and above their wall.
  - A *Group* gadget can be added as a selectable list item in the left panel area of a group and is rendered in its own single item view.
  - A *Statusbar* gadget renders on the bottom of every screen as an interactive button that can show alerts, badges and expand into a collapsible window.
- **Gadget instantiation:**
  - *Content* gadgets act like any other group content item. After an admin registers one or more gadgets, members of the company can create one or more instances of each gadget within the groups of the company. Each instance belongs to the group to which it was added and therefore all group members share that instance and its data. Each instance of the gadget has its own data store.
  - For *Profile* gadgets, a non-admin user has no control over adding a gadget instance to his or her profile. A profile gadget instance cannot be created for some users in the company and not others. A profile gadget instance is added automatically after the company admin registers and enables it upon the first refresh of each user's profile page in that company. After an admin registers one or more gadgets, members of the company automatically get **one** instance per gadget that shows up on their profile page. As a result, each user has a separate gadget instance that has its own data storage. There is only one instance of the registered gadget on the profile page of each user. Users don't share the instance or its data.
  - *Group* gadgets are similar to content gadgets with two differences. Only one group gadget can be assigned to a group and only group admins can add them to a group. Each instance belongs to the group to which it was added and therefore all group members share that instance and its data. Each instance of the gadget has its own data store.
  - For *Statusbar* gadgets, a non-admin user has no control over adding a gadget instance to the bottom of the screen. A statusbar gadget instance cannot be created for some users in the company and not others. Each instance is added automatically after the company admin registers and enables it upon the first refresh of every page in that company. As a result, each user has a separate gadget instance on the bottom of each screen. Users don't share the instance or its data. Each instance of the gadget does not have its own data store.
- **Gadget deletion privileges:**

- *Content* gadgets act like any other group content item. If a user has privileges to delete a content item, they can delete a content gadget. If a user deletes a gadget from a group's Content section, only that instance of the gadget will be affected; other instances of the gadget will continue to work and be displayed until they are deleted.
- For *Profile* gadgets, non-admin users have no control over deleting a gadget instance from their profile page. A profile gadget instance cannot be deleted or disabled for some users in the company and not others. When a company admin deletes a profile gadget, it is deleted for *all* users.
- *Group* gadgets cannot be removed from the groups they have been added to. Once they are added to a group, they are permanently part of that group.
- For *Statusbar* gadgets, non-admin users have no control over deleting a gadget instance. A statusbar gadget instance cannot be deleted or disabled for some users in the company and not others. When a company admin deletes a statusbar gadget, it is deleted for *all* users.
- **Gadget deletion behavior:**

Once a company administrator disables or deletes a gadget in the [Admin](#) [Integrations](#) [OpenSocial Gadgets](#) page, all instances of that gadget cease to function:

  - *Content* gadgets render an error message.
  - *Profile* gadgets disappear from the users' profile pages.
  - *Group* gadgets render an error message.
  - *Statusbar* gadgets disappear from the bottom of every screen.
- **View:** The OpenSocial API has a feature that allows the developer to run different code depending on the "view" value; in SAP Jam gadgets, the view value can be set to:
  - "home" for *Content* gadgets (view="home")
  - "profile" for *Profile* gadgets (view="profile")
  - "home" for *Group* gadgets (view="home")
  - "home" for *Statusbar* gadgets (view="home")
- **Gadget saved data:**
  - *Content* gadgets can save data and will not lose saved data when they are disabled.
  - *Profile* gadgets can save data and will lose saved data when they are disabled.
  - *Group* gadgets can save data and will not lose saved data when they are disabled.
  - *Statusbar* gadgets cannot save data.

## 4.1.2 Gadget Content Types

This page describes the URL and HTML Gadget Content Types available to SAP Jam Collaboration OpenSocial Gadgets.

SAP Jam Collaboration uses URL and HTML OpenSocial Gadget Content Types to render web content in the gadget. The gadget content type must be specified in the gadget's XML within the <Module specificationVersion='2'> root element as shown below:

```
<Module specificationVersion='2'>
  <Content type = "[gadget_content_type]">
    </Content>
</Module>
```

## URL Gadget Content Type

URL Gadgets run content directly from a URI in the gadget. The gadget content type must be specified in the gadget's XML within the `<Module specificationVersion='2'>` root element with the URI as shown below:

```
<Module specificationVersion='2'>
  <Content type = "url" href="[content_URI]">
  </Content>
</Module>
```

## HTML Gadget Content Type

HTML Gadgets run web content directly embedded in the gadget's XML or from a URI.

To run web content directly embedded in the gadget the HTML/Javascript code must be enclosed in `<![CDATA[ ... ]]>` tags within the gadget content type tags within the `<Module specificationVersion='2'>` root element as shown below:

```
<Module specificationVersion='2'>
  <Content type = "html">
    <![CDATA[
      <HTML_and_Javascript_code>
    ]]>
  </Content>
</Module>
```

A HTML gadget tutorial that includes source code and a ready to use gadget is available [here \[page 87\]](#).

To run web content from a URI the gadget content type must be specified in the gadget's XML within the `<Module specificationVersion='2'>` root element with the URI as shown below:

```
<Module specificationVersion='2'>
  <Content type = "html" href="[content_URI]">
  </Content>
</Module>
```

A HTML gadget tutorial that includes source code and a ready to use gadget is available [here \[page 92\]](#).

## 4.1.3 Register your gadget with SAP Jam Collaboration

This page documents the steps required to register your OpenSocial gadget with SAP Jam Collaboration.

OpenSocial gadgets provide a mechanism to extend SAP Jam Collaboration's abilities to interact with any software or service.

SAP Jam Collaboration provides access to pre-built, third-party OpenSocial gadgets that are ready to use by simply enabling them. You can also download some existing OpenSocial gadgets from the [SAP Jam Sample Code GitHub](#) site's [OpenSocial Gadgets](#) section. There is also support for your organization to develop

your own OpenSocial gadgets, and to make those available to your SAP Jam users. For information on how to add, enable, and manage OpenSocial gadgets, please see the following:

- [To import an OpenSocial gadget configuration \[page 75\]](#) explains how to upload a gadget configuration file.
- [To add an OpenSocial gadget \[page 75\]](#) explains how to register a gadget that does not have a configuration file.
- [To enable an OpenSocial gadget \[page 81\]](#) explains the various ways in which you can enable a gadget and it provides [an explanation of the warning that is displayed when you first enable a particular OpenSocial gadget \[page 81\]](#).
- [To manage OpenSocial gadgets \[page 81\]](#) explains how to view, edit, or delete an OpenSocial gadget.

## To import an OpenSocial gadget configuration

You can upload a gadget configuration file for any OpenSocial gadget that you download from the [SAP Jam Sample Code GitHub](#) site's [OpenSocial Gadgets](#) section. Gadget configuration files are the files that you get when you click on **Export Gadget Configuration** in a gadget's administration page. To import an OpenSocial gadget's configuration file, do the following:

1. From the [Integrations > OpenSocial Gadgets](#) page, click [Import Gadget Configuration](#) near the top of the page.  
Your browser will open a file upload dialog box.
2. Navigate to the configuration file on your device and upload it.  
After importing the file, the [Register OpenSocial Gadget](#) page displays with the file's configuration information shown in the form.
3. Change the existing settings or fill in any missing settings as required and click [Save](#).  
You are returned to the [OpenSocial Gadgets](#) page, with the gadget that you registered listed in the catalog.

## To add an OpenSocial gadget

For OpenSocial gadgets that you have developed within your own organization, you must fill in the [Register OpenSocial Gadget](#) form, as described in the following:

1. In SAP Jam Collaboration, as a company administrator, click on the cog icon at the top of the page, select [Admin](#) from the context menu, and select [Integrations > OpenSocial Gadgets](#) from the left navigation sidebar.  
The [OpenSocial Gadgets](#) page displays.


## OpenSocial Gadgets (Company)

+ Add Gadget

↑ Import Gadget Configuration

Third Party Gadgets

Custom Gadgets




### SurveyMonkey

Author: SAP

SurveyMonkey is the world's leading online survey platform, with more than 3 million survey responses every day. SurveyMonkey has revolutionized the way people give and take feedback, making it accessible, [More...](#)

Context: Content

Enabled: ☒
[View](#)



### BrainShark

Author: SAP

Brainshark sales enablement software accelerates revenue through faster training, increased demand, and more successful sales conversations.

Context: Content

Enabled: ☒
[View](#)

#### OpenSocial Gadgets catalog

- Click [Add Gadget](#) at the upper right corner of the page.  
The [Register OpenSocial Gadget](#) page displays.



## Register OpenSocial Gadget (Company)

---

### Gadget Configuration

---

URL\*

Enabled ☐

Context

Search Paths

OAuth 1.0a Service Configurations

Service Name	Consumer Key	Signature Method
<a href="#">Add Service Configuration</a>		

---

OAuth 2.0 Service Configurations

Service Name	Client Id	Grant Type
<a href="#">Add Service Configuration</a>		

---

Administrative Area

Gadget Information  Choose Display Language:

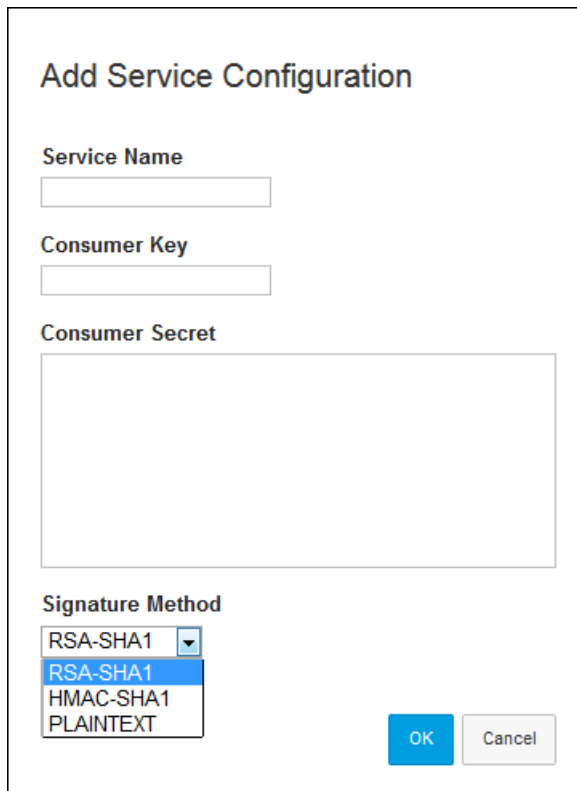
The Register OpenSocial Gadget form

- In the [URL](#) field, enter the URL of the OpenSocial gadget that you want to register.
- Optionally, select the [Enabled](#) checkbox to have the gadget immediately available.  
The gadget can be enabled at any time, but be certain that you are aware of implications of doing so before you enable it. This information is shown in the note in the procedure [To enable an OpenSocial gadget \[page 81\]](#).
- From the [Context](#) drop-down menu, select either: [Admin](#) > [OpenSocial Gadgets](#) > [Register OpenSocial Gadget](#) page. The URL must point to the XML file for your gadget.  
When you register your SAP Jam OpenSocial Gadget-compliant application, you can configure it to have one of four types of [Contexts](#):
  - Content:** Content gadgets can be added to a group's Content section by anyone in the company with the appropriate privileges. They appear as an addable option in the Content area of SAP Jam Collaboration groups.
  - Profile:** Profile gadgets appear on the profile of each member of the company automatically. They appear in the [Product Setup](#) > [Custom Profile](#) page, where you can configure where they will appear in your users' profile pages.
  - Group:** Group gadgets can be added to a group by anyone in the company with the appropriate privileges. They appear as a selectable list item in the left panel area of a SAP Jam Collaboration group.

- Select [Group Admin](#) > [Edit Group](#) > [Setup](#) > [Select Primary Group Extension](#) to add a group gadget to a group.
  - **Note:** Only one group gadget can be added to a group and it cannot be removed from the group it has been added to.
  - **Statusbar:** Statusbar gadgets can be added by anyone in the company with the appropriate privileges. They appear on the bottom of every screen as an interactive button that can show alerts, badges and expand into a collapsible window.
6. Optionally, to configure access to the data stores for personalized data for the gadget, you can enter a list of parameters in the [Search Paths](#) field.
- These parameters allow you to descend into the AppData pool and the public\_wave pool JSON data structures. Search parameters can be space- or comma-delimited, and wildcards can be set on the JSON keys. For example:
- **a** — This example searches for all data within **a** keys.
  - **a.b** — This example searches for all data within **b** keys that are within **a** keys.
  - **a.b\*** — This example uses \* as a wildcard to search for all data within keys that start with **b** (using the \* wildcard) that are within **a** keys.
  - **a.b\* c.d** — This example uses a space delimiter to provide two search paths. The first search path (**a.b\***) searches for all data within keys that start with **b** that are within **a** keys. The second search path searches for all data within **c** keys that are within **d** keys.
  - **a.b\*,c.d** — This example uses a comma delimiter rather the space delimiter used in the previous example; otherwise, it performs the same searches as the previous example.
7. Optionally, to enable access to the gadget and its displayed data using OAuth 1.0a, click [Add Service Configuration](#) in the [OAuth 1.0a Service Configurations](#) section.
- An [Add Service Configuration](#) dialog box displays.

In the OAuth 1.0a Add Service Configuration dialog box, set the following:

1. In the [Service Name](#) text box, enter the service name for your OAuth 1.0a service.
2. In the [Consumer Key](#) text box, enter the consumer key for your OAuth 1.0a service.
3. In the [Consumer Secret](#) text box, enter the consumer secret for your OAuth 1.0a service.
4. From the [Signature Method](#) drop-down menu, select the consumer secret for your OAuth 1.0a service.
5. Click [OK](#) to save these options, or click [Cancel](#) to abandon them.  
You are returned to the [OpenSocial Gadgets](#) page.

The image shows a dialog box titled "Add Service Configuration". It contains four main sections: "Service Name" with a single-line text input; "Consumer Key" with a single-line text input; "Consumer Secret" with a multi-line text area; and "Signature Method" with a drop-down menu. The drop-down menu is currently open, showing four options: "RSA-SHA1" (which is highlighted), "RSA-SHA1", "HMAC-SHA1", and "PLAINTEXT". At the bottom right of the dialog box are two buttons: "OK" and "Cancel".

**Add Service Configuration**

**Service Name**

**Consumer Key**

**Consumer Secret**

**Signature Method**

RSA-SHA1 ▼

RSA-SHA1

HMAC-SHA1

PLAINTEXT

[OK](#) [Cancel](#)

8. Optionally, to enable access to the gadget and its displayed data using OAuth 2.0, click [Add Service Configuration](#) in the [OAuth 2.0 Service Configurations](#) section.  
An [Add Service Configuration](#) dialog box displays.

In the OAuth 2.0 Add Service Configuration dialog box, set the following:

1. In the [Service Name](#) text box, enter the service name for your OAuth 2.0 service.
  2. In the [Client Id](#) text box, enter the client ID for your OAuth 2.0 service.
  3. In the [Client Secret](#) text box, enter the consumer secret for your OAuth 2.0 service.
  4. Select the authorization [Grant Type](#) that you want to use.
  5. If you chose the [SAML 2.0 Bearer Assertion](#), then you must also select the [SAML 2.0 Assertion Audience](#).
  6. Click [OK](#) to save these options, or click [Cancel](#) to abandon them.
- You are returned to the [OpenSocial Gadgets](#) page.

**Add Service Configuration**

**Service Name**

**Client Id**

**Client Secret**

**Grant Type**  
☐ Authorization Code (authorization\_code)  
☒ SAML 2.0 Bearer Assertion (urn:ietf:params:oauth:grant-type:saml2-bearer)

**SAML 2.0 Assertion Audience**  
☒ Token endpoint URL of the authorization server  
☐ Custom

[OK](#) [Cancel](#)

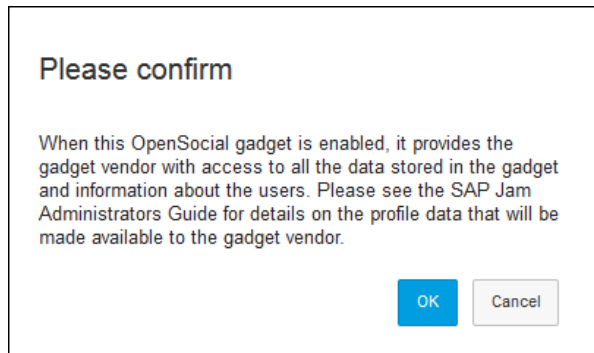
9. From the [Administrative Area](#) drop-down menu, select the administrative area in which you want this gadget to be available. If you choose a specific area, the gadget will only be available for use in the groups in that area. If you choose "Company", the gadget will be available for use in all of the groups in your organization.
10. From the [Choose Display Language](#) drop-down menu, select the language that you want used in the OpenSocial gadget.
11. At any time after you have entered the URL for the gadget, you can click [Refresh](#) beside [Gadget Information](#) to view important information about the gadget and to see a preview of the gadget.
12. When all of the above settings are complete and to your satisfaction, click [Save](#) to register the OpenSocial gadget for use in SAP Jam Collaboration.  
You are returned to the [OpenSocial Gadgets](#) page, with the gadget that you registered listed in the [Custom Gadgets](#) tab's catalog.

## To enable an OpenSocial gadget

You can enable an OpenSocial gadget for use by the members of your organization's SAP Jam instance in several ways:

You can enable an OpenSocial gadget in the following ways:

- Turn on the *Enabled* switch in the row for the gadget in the *Third Party Gadgets* tab.
- Turn on the *Enabled* switch in the row for the gadget in the *Custom Gadgets* tab.
- Select the *Enabled* option in the *Register OpenSocial Gadget* page.
- Turn on the *Enabled* switch in the *View* gadget page.



The first time that you enable a particular OpenSocial gadget in your SAP Jam instance, the confirmation dialog box shown above will display. See the note below for an explanation of the warning about the profile data that will be made available to the gadget vendor.

### Note

For any OpenSocial gadget, there may be risks involved in the exposure of data to external sources. Any data that is stored for use in the operation of the gadget in an external data store will be available to that external user or organization. For example, marking a location on a Google Map stores the data related to that marker that can be accessed by Google. Similarly, if a third-party survey gadget is added, the organization that provided the gadget will have access to the responses to the survey.

Also, certain personal data can be exposed, such as:

- The user's first and last name
- The user's email address
- The user's thumbnail image or avatar

The scope of this personal information depends on the context of the gadget, specifically:

- **Content gadgets:** can expose data on all members of the group to which they have been added.
- **Profile gadgets:** can expose data on all members that the user who has added the gadget is following.

Please consider the security implications of this data exposure for any OpenSocial gadget that you add to SAP Jam.

## To manage OpenSocial gadgets

### To view the configuration of a gadget

You can view the configuration of either a "Third Party Gadget" or a "Custom Gadget" by doing the following:

1. In either tab of the [OpenSocial Gadgets](#) page, click [View](#) on the row for the OpenSocial gadget whose configuration information you want to see.

An [OpenSocial Gadget: <gadget\\_name>](#) page displays.

**SAP Lumira URL Gadget**

OpenSocial Gadgets

Edit Reload Gadget Source Export Gadget Configuration

**Gadget Configuration**

**URL:** `https://sapjamsamplecode.github.io/OpenSocial/Gadget/HCP_Lumira/HCP_Lumira.xml`

**Context:** Content

**Enabled:** ☒

**Search Paths:**

**OAuth 1.0a Service Configurations:** None

**OAuth 2.0 Service Configurations:** None

**Gadget Information** Choose Display Language: English

An OpenSocial Gadget view page

Note that the gadget information and preview sections are not shown.

2. Optionally, to edit the gadget's configuration, click [Edit](#) near the top of the page.
3. Optionally, to exit the [View](#) page, click [OpenSocial Gadgets](#) near the top of the page.  
You are returned to the [OpenSocial Gadgets](#) page.
4. Note that you can [Enable](#) the gadget from this page. If you chose to do so, [be certain that you are aware of implications of doing so \[page 81\]](#) before you do.

### To modify the configuration of a gadget

Note that you cannot modify [Third Party Gadgets](#).

You can edit a "Custom Gadget" by doing the following:

1. In the [Integrations](#) > [OpenSocial Gadgets](#) page, click on the [Custom Gadgets](#) tab.  
The OpenSocial Gadgets [Custom Gadgets](#) tab displays.

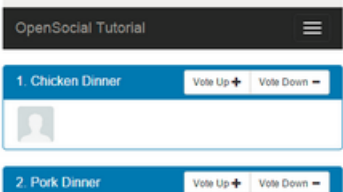
## OpenSocial Gadgets (Company)

+ Add Gadget

Import Gadget Configuration

Third Party Gadgets

Custom Gadgets



### React Tutorial

Author: Geon Young


SAP JAM Tutorial App Built with React

Context: Content

Date Added: June 01, 2015 15:33

Enabled: ☐

[View](#) | [Edit](#) | [Delete](#)



### Lumira


Author:

Context: Content

Date Added: January 19, 2015 15:01

Enabled: ☒

[View](#) | [Edit](#) | [Delete](#)



### To-Do List

Author: Labpaxies

Easily manage and track everything you need To-Do. The gadget lets you create multiple To-Do lists, each with a unique purpose. Use To-Do List to create a list of your daily tasks, plan the required party [More...](#)

Context: Content

Date Added: March 24, 2015 14:44

Enabled: ☐

[View](#) | [Edit](#) | [Delete](#)

The OpenSocial Gadgets Custom Gadgets tab

Note that the gadget information and preview sections are not shown.

- Find the custom gadget that you want to modify and click [Edit](#) on the row for that gadget to modify its configuration.

The [Edit OpenSocial Gadget](#) page displays.

## Edit OpenSocial Gadget

---

### Gadget Configuration

---

URL\*

Enabled ☒

Context

Search Paths

OAuth 1.0a Service Configurations

Service Name	Consumer Key	Signature Method
<a href="#">Add Service Configuration</a>		

---

OAuth 2.0 Service Configurations

Service Name	Client Id	Grant Type
<a href="#">Add Service Configuration</a>		

---

Administrative Area

Gadget Information  Choose Display Language:

#### Edit an OpenSocial Gadget configuration

The *Edit* page is effectively identical to the *Register OpenSocial Gadget* page (so you can perform most of the steps in [To add an OpenSocial gadget \[page 75\]](#)), although you cannot modify the *Context* once it is set.

3. Make whatever changes are required.
4. Click [Save](#) to save your changes.

You are returned to the *OpenSocial Gadgets* page, with the modified OpenSocial Gadget that you just edited listed in the catalog.

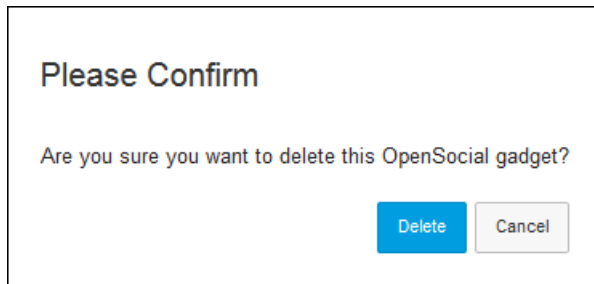
#### To delete a Custom Gadget

Note that you cannot delete *Third Party Gadgets*.

You can delete a "Custom Gadget" by doing the following:

1. In the [Integrations](#) > [OpenSocial Gadgets](#) page, click on the *Custom Gadgets* tab. The OpenSocial Gadgets *Custom Gadgets* tab displays.
2. Find the custom gadget that you want to modify and click [Delete](#) on the row for that gadget to modify its configuration. A delete confirmation dialog box displays.





Confirmation to delete an OpenSocial Gadget

3. Click [Delete](#).

You are returned to the [Custom Gadgets](#) tab, where the selected gadget has been removed from the catalog.

## 4.1.4 OpenSocial Gadget XML Structure

This page describes the XML markup and structure of OpenSocial gadgets.

The basic structure of your required OpenSocial gadget XML markup is shown in the following example.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Module specificationVersion='2'>
  <ModulePrefs title="hello world"/>
  <Content type="html">
    <![CDATA[Hello, world!]]>
  </Content>
</Module>
```

A gadget's XML must be contained within a `<Module specificationVersion='2'>` root element. This element contains the following child elements:

- `<ModulePrefs>` (*required*)—discussed in detail below.
  - `<Locale>` (*optional*)—discussed in detail below.
- `<UserPrefs>` (*optional*)—discussed in detail below.
- `<Content>` (*required*)—must be set with an attribute of "type" that has a value of "html", which indicates that the gadget's content is in HTML: `<Content type="html">`.

Within the content tags, the HTML and JavaScript are enclosed in `<![CDATA[ ... ]]>` tags to indicate to the gadget parser that the enclosed text should not be parsed as XML.

A more fully developed version of the "Hello World" example of an OpenSocial gadget's XML markup is shown in the following example.

```
<Module specificationVersion='2'>
  <ModulePrefs
    title="Hello World example"
    description="Sample Hello World gadget"
    author="Test User"
    thumbnail="https://example.com/images/thumbnail.png">
    <Require feature="dynamic-height" />
  </ModulePrefs>
  <UserPref
    name="mycolor"
    display_name="Color"
    default_value="Lime"
```

```

        datatype="string" />
<Content type="html">
  <![CDATA[
    <script type="text/JavaScript">
      // Get the preferences for the current user
      var prefs = new gadgets.Prefs();
      function hello_world() {
        var html="Hello World!";

document.getElementById("content_div").style.backgroundColor=prefs.getString("mycolor");

        document.getElementById("content_div").innerHTML = html;
        gadgets.window.adjustHeight();
      }
      gadgets.util.registerOnLoadHandler(hello_world);
    </script>
    <div id="content_div"></div>
  ]]>
</Content>
</Module>

```

The significant XML elements and attributes in the above example are discussed below.

- **<ModulePrefs>** (*required*)—contains information that describes the gadget. It has the following attributes:
  - **title**="[value]" (*required*)—A string that provides the title of the gadget as it appears in the SAP Jam Collaboration group or user profile to which it has been added.
  - **description**="[value]" (*optional*)—A string that provides a more verbose description of the gadget than the title.
  - **author**="[value]" (*optional*)—A string containing the name of the author of the gadget.
  - **thumbnail**="[value]" (*optional*)—An optional string that provides the URL of the gadget thumbnail.
  - **<Require>** (*optional*)—specifies the required API feature(s) for the gadget.
- **<UserPrefs>** (*optional*)—defines controls that allow users to specify their own settings for the gadget. For example, a restaurant recommendation gadget would benefit from the ability to store user's location to find the closest restaurants.
  - **name**="[value]" (*required*)—name of the user preference which serves as a key to retrieve the preference.
  - **default\_value**="[value]" (*optional*)—the default value of the preference.
  - **datatype**="[value]" (*optional*)—a string to indicate a data type of the attribute.

In "Hello World 2" we used UserPrefs to remember the color that we want to use for the background. An app accesses the UserPrefs by calling `gadgets.Prefs()`. To get the value of a specific user preference, in this example, the "mycolor" preference, call `prefs.getString("mycolor")`.
- **<Content type="html">** (*required*)—contains the JavaScript and HTML that is required for the gadget to run. A gadget's JavaScript and HTML must be enclosed within a `[CDATA[ ... ]]` directive so that the content isn't parsed as XML.

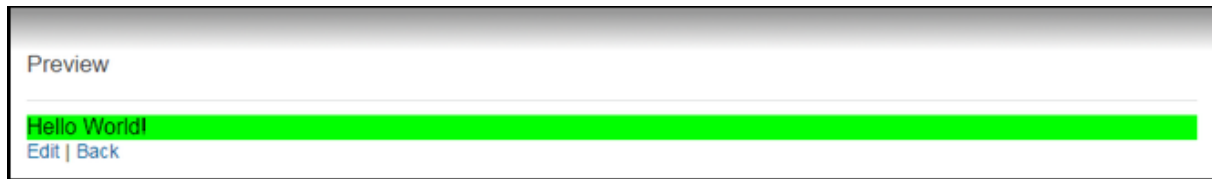
To begin initialization, call:

```

gadgets.util.registerOnLoadHandler(function() {
  your code here;
});

```

The code example above produces the following gadget, as shown in the [Admin > OpenSocial Gadgets > Add OpenSocial Gadget](#) page's *Preview* pane:



### Note

There are no specific requirements for the HTML used in your gadget, although any JavaScript code must be given appropriate HTML markup with which it will be displayed. Beyond that requirement, you can write as much or as little HTML content to be a part of your OpenSocial gadget as you want.

## References

- See [Lesson 1 - Hello World \[page 90\]](#) for the SAP Jam OpenSocial gadget tutorial page on the XML basics.
- See the Google [Gadgets XML Reference](#) page for more details on the XML option for you gadget.
- See the [Lesson 2 - UI Setup \[page 92\]](#) page for the SAP Jam OpenSocial gadget tutorial page on setting up the HTML and, optionally, using CSS styling.
- See the Google Developers <https://developers.google.com/gadgets/docs/ui> page to find out more about OpenSocial gadgets UI (HTML) options.

## Next

With the required XML and HTML set, proceed to [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#) to develop the JavaScript code for your SAP Jam OpenSocial gadget.

## 4.1.5 SAP Jam Collaboration OpenSocial Tutorials

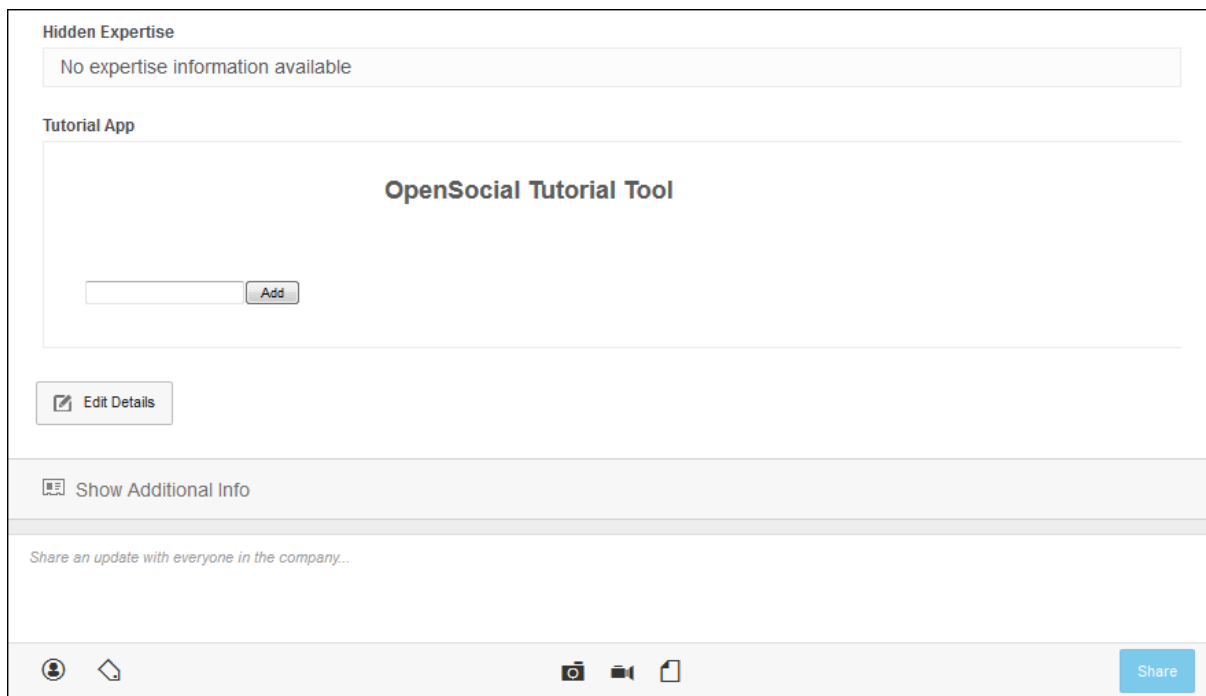
This section provides tutorials for developing SAP Jam Collaboration OpenSocial gadgets.

### 4.1.5.1 Hello World Gadget Tutorial

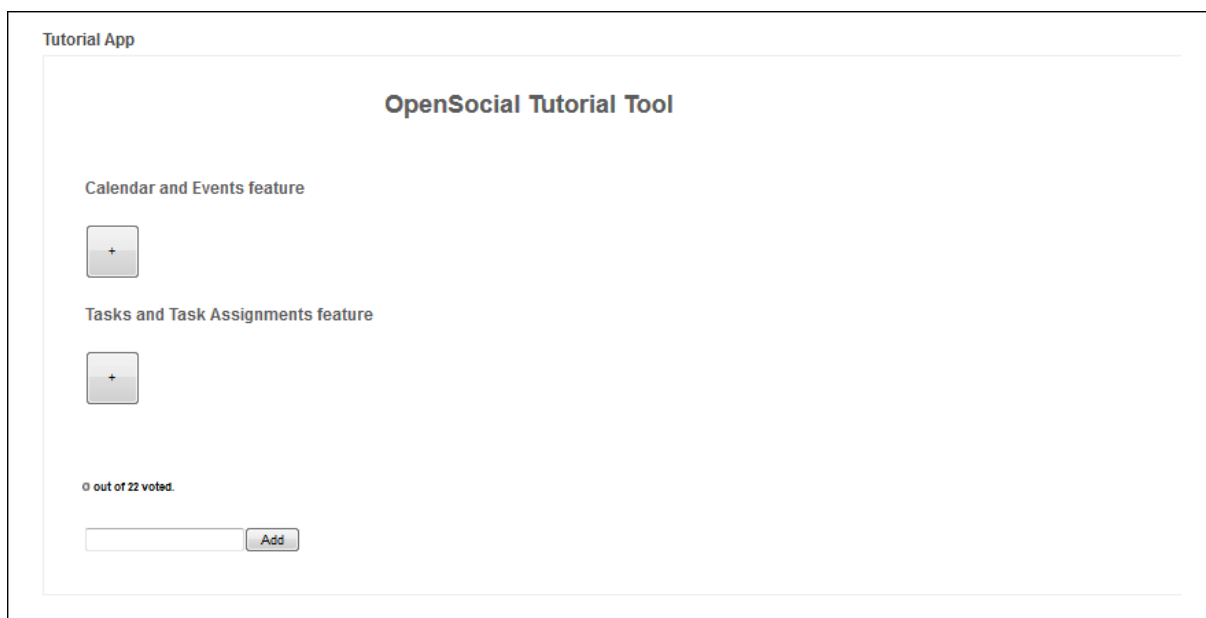
This section provides a quick tutorial for developing a simple SAP Jam Collaboration OpenSocial "Hello World" gadget and a "Voting Tool" gadget.

In this tutorial we will be building these 2 basic gadgets in 7 lessons. In the first lesson we will create the "Hello World" gadget to use a single XML file to print text to the screen.

In the remaining lessons we will create the "Voting Tool" gadget which enables users to publicly vote on the assignment of tasks. Initially, the gadget will appear without any topics listed. Note that this tutorial example gadget has been registered with SAP Jam as a "Content" gadget.

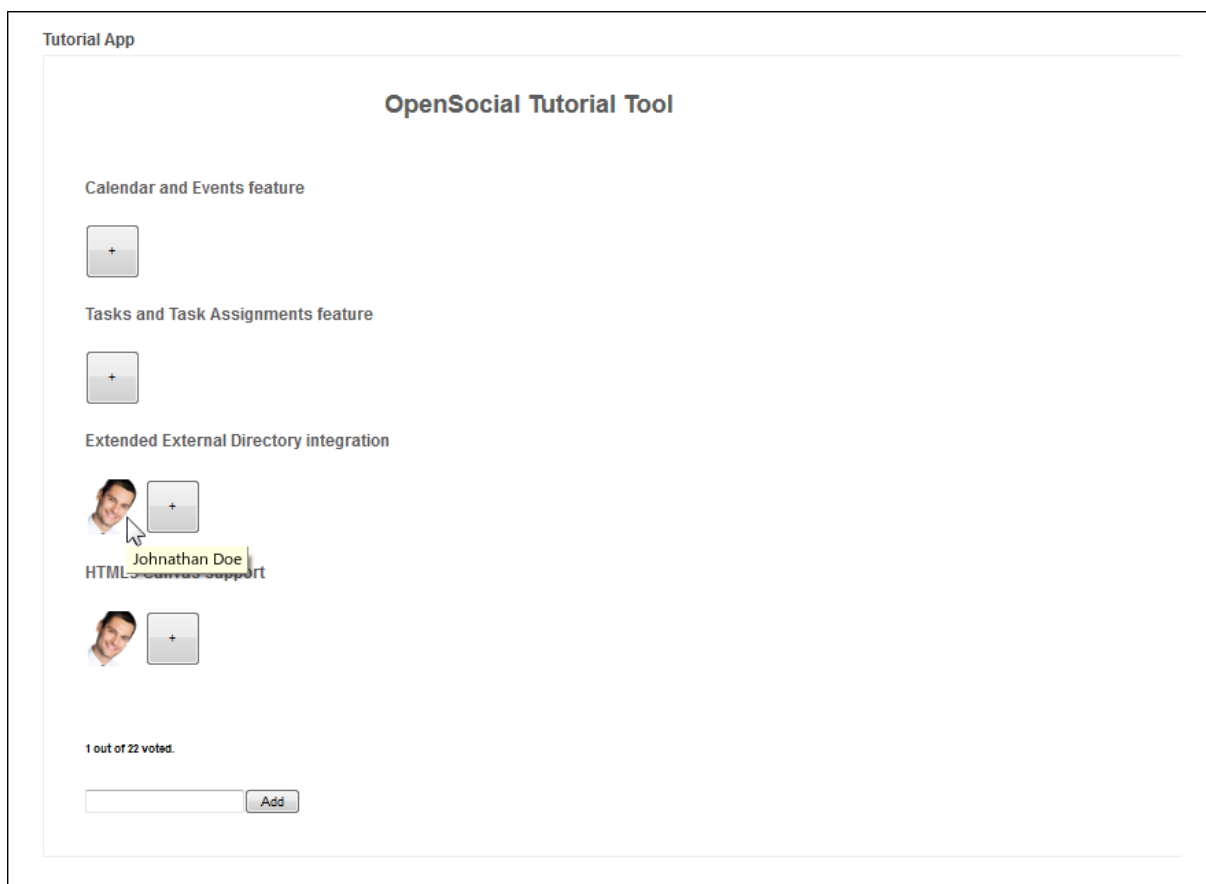


Any user will be able to add more topics at any time. As they do, the topic titles will appear with a "+" voting button displayed underneath the title.

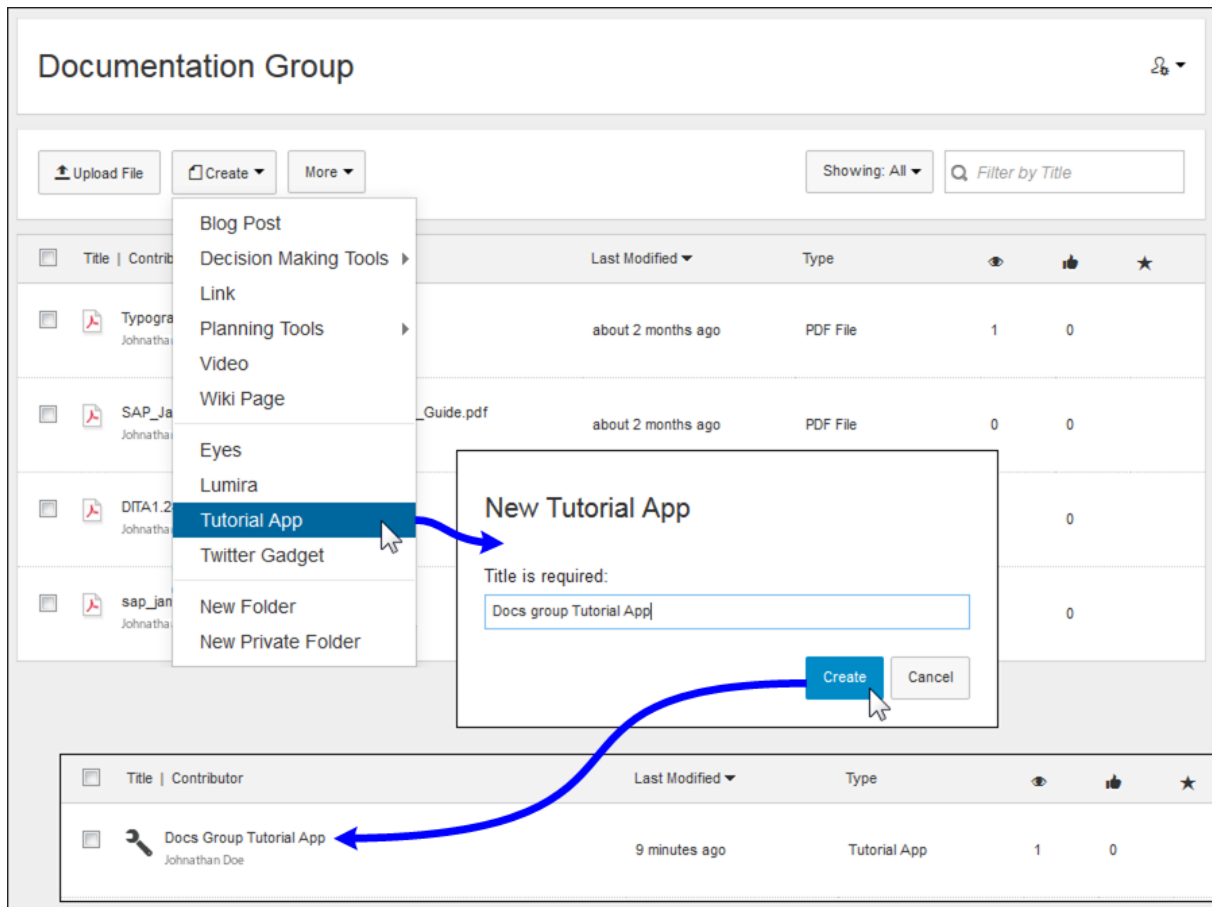


Also, any user will be able to "vote" for any or all of the options (volunteer for a task in a scenario of team allocations to a set of tasks). Once the user clicks on a "+" vote button, that voter's profile thumbnail will appear under that topic, to the left of the vote button. Subsequently, any group member can hover over the thumbnails to see the voter's name. Also, near the bottom of the gadget, the number of members that have

participated in the poll will be displayed along with a count of all group members. This gadget will update in real time with no need for page refreshes to get data changes.



If we register this gadget with SAP Jam as a "Content" gadget, the gadget title, as specified in ModulePrefs (discussed later) will appear in SAP Jam's **Content > Create** menu. If you add a new gadget instance, you are prompted to give it a title, and the gadget instance then appears in a new Content page under the given title.



### Note

The code in this tutorial is for demo purposes only. It sacrifices robustness for code simplicity to better focus on the core elements.

## SAP Jam Beginner Gadget Tutorial

The pages in this tutorial are:

### 4.1.5.1.1 Lesson 1 - Hello World

This first lesson of the tutorial shows you how to set up a basic "Hello World" sample gadget that will demonstrate both the initial XML markup and a pre-made sample you can run directly in SAP Jam Collaboration.

Let's begin with a "Hello, world!" gadget to see the basic building blocks of any OpenSocial gadget:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Tutorial_01_Hello_World (XML) -->
```

```
<Module specificationVersion='2'>
  <ModulePrefs title="Tutorial_01_Hello_World">
  </ModulePrefs>
  <Content type="html">
    <![CDATA[
      Hello, world!
    ]]>
  </Content>
</Module>
```

A gadget's XML must be contained within a `<Module specificationVersion='2'>` root element. This element contains the following child elements:

- `<ModulePrefs>` (*required*)—discussed in detail below.
- `<UserPrefs>` (*optional*)—discussed in detail below.
- `<Content>` (*required*)—must be set with an attribute of "type" that has a value of "html", which indicates that the gadget's content is in HTML: `<Content type="html">`.

Within the content tags, the HTML and JavaScript are enclosed in `<![CDATA[ ... ]]>` tags to indicate to the gadget parser that the enclosed text should not be parsed as XML.

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of `hello_world.xml` with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_01\\_Hello\\_World/hello\\_world.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_01_Hello_World/hello_world.xml)



Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam Collaboration live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

## Next

Proceed to [Lesson 2 - UI Setup \[page 92\]](#).

## 4.1.5.1.2 Lesson 2 - UI Setup

This second lesson of the tutorial shows you a basic setup of the HTML in which your gadget will be displayed, as well as some style options to demonstrate that CSS can be used to control the appearance of your gadget.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> 📄

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open ui\_setup.xml. This file is an exact copy of hello\_world.xml that has had the title renamed to "Tutorial\_02\_UI\_Setup" and height adjusted to 250 pixels as shown below:

```
<!--
* STEP 1
* Let's rename our application and set a default height for the gadget to 250 in
ModulePrefs
-->
<ModulePrefs title="Tutorial_02_UI_Setup" height="250">
</ModulePrefs>
```

Open ui\_setup.html to view the basic HTML structure that has been added to our gadget as shown below:

```
<!--
* STEP 2
* Add some basic HTML structure to our gadget
-->
<div id="app">
  <div id="header"><h2>OpenSocial Tutorial Tool</h2></div>
  <div id="body"></div>
  <div id="footer"></div>
</div>
```

Open ui\_setup.css to view the CSS styling which has been added to our gadget as shown below:

```
/*
* STEP 3
* Add CSS styling to set properties for app, header, body and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

ui\_setup.css is externally linked to ui\_setup.html as shown below:

```
<!--
*STEP 3
* Add CSS styling to set properties for app, header, body and footer elements.
-->
<link rel="stylesheet" type="text/css" href="ui_setup.css">
```



ui\_setup.html is externally linked to ui\_setup.xml as shown below:

```
<Content type="html" href="ui_setup.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of ui\_setup.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_02\\_UI\\_Setup/ui\\_setup.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_02_UI_Setup/ui_setup.xml) 📄

Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> 📄 .

## Next

Proceed to [Lesson 3 - Initialize \[page 93\]](#).

## 4.1.5.1.3 Lesson 3 - Initialize

This third lesson of the tutorial shows you how to initialize and render JavaScript for your gadget.

While the [Lesson 1 - Hello World \[page 90\]](#) page demonstrated the XML markup, and the [Lesson 2 - UI Setup \[page 92\]](#) page demonstrated the HTML markup, the majority of the work on any gadget will be in JavaScript.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> 📄

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open initialize.xml. This file is an exact copy of ui\_setup.xml that has had the title renamed to "Tutorial\_03\_Initialize" and added `<Require feature="wave" />` in `<ModulePrefs>` as shown below:

```
<!--
* STEP 1
* - Rename our application
* - Add "wave" as a required feature to ModulePrefs
-->
<ModulePrefs title="Tutorial_03_Initialize" height="250">
  <Require feature="wave"/>
</ModulePrefs>
```

To use wave in our gadget, the wave framework has been added to ModulePrefs. The wave framework will allow for real-time updates, storage, and information retrieval about the users of the gadget. You will also notice other changes in initialize.xml. Ignore them for now, we will discuss them further into the tutorial.

Open initialize.css. It is an exact copy of ui\_setup.css and contains all of the CSS styling for our gadget as shown below:

```
/* Add CSS styling to set properties for app, header, body, and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

initialize.css is externally linked to initialize.html as shown below:

```
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="initialize.css">
```

Open initialize.js. It contains all of the Javascript used in our gadget. Take note of Steps 2 and 3 in the source code shown below:

```
/**
* STEP 2 - Add Javascript that sets callbacks and renders the gadget.
*/
// Renders the gadget
function renderInfo() {
}
// Sets callbacks
function init() {
  if (wave && wave.isInWaveContainer()) {
    // Loads the gadget's initial state and the subsequent changes to it
    wave.setStateCallback(renderInfo);
    // Loads participants and any changes to them
    wave.setParticipantCallback(renderInfo);
  }
}
/**
* STEP 3
* Initialize our gadget by passing init().
*/
```

```
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);
```

initialize.js is externally linked to initialize.html as shown below:

```
<script type="text/javascript" src="initialize.js"></script>
```

initialize.html is externally linked to initialize.xml as shown below:

```
<Content type="html" href="initialize.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of initialize.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_03\\_Initialize/initialize.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_03_Initialize/initialize.xml) ➡

Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> 📖 ➡ .

## Next

Proceed to [Lesson 4 - Adding topics \[page 95\]](#).

## 4.1.5.1.4 Lesson 4 - Adding topics

This fourth lesson of the tutorial expands on the previous gadget by allowing users to dynamically add "topics" at any time. It demonstrates the use of some of the `gadget.json` functions to store an array of strings, and also demonstrates the use of some of the Apache wave functions to ensure that changes made by others to this array of JSON strings are updated in real time to keep the list of options that we are encoding current for all users.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> ➡

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open `adding_topics.xml`. This file is an exact copy of `initialize.xml` that has had the title renamed to "Tutorial\_04\_Adding\_topics" and added `<Require feature="dynamic-height" />` in `<ModulePrefs>` as shown below:

```
<!--
* STEP 1
* - Rename our application
* - Add "dynamic-height" as a feature to ModulePrefs
-->
<ModulePrefs title="Tutorial_04_Adding_topics" height="250">
  <Require feature="wave"/>
  <Require feature="dynamic-height"/>
</ModulePrefs>
```

This dynamic-height feature will allow the window to resize dynamically as topics are added. You will also notice other changes in `adding_topics.xml`. Ignore them for now, we will discuss them further into the tutorial.

Open `adding_topics.css`. It is an exact copy of `ui_setup.css` and contains all of the CSS styling for our gadget as shown below:

```
/* Add CSS styling to set properties for app, header, body and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

`adding_topics.css` is externally linked to `adding_topics.html` as shown below:

```
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="adding_topics.css">
```

Open `adding_topics.js`. It contains all of the Javascript used in our gadget. Take note of Steps 2 and 3 in the source code and all content marked with `/**` and `*/` shown below:

```
/**
* STEP 2 (Implementing the "Add Topic" button)
* - Encode object as JSON string -> toJSON(obj)
* - Decode JSON string into an object -> toObject(str)
* - Stores data in the OpenSocial gadget -> addInput()
*/
/** Encode object as JSON string */
function toJSON(obj) {
  return gadgets.json.stringify(obj);
}
/** Decode JSON string into an object */
function toObject(str) {
```

```

        return gadgets.json.parse(str);
    }
    /** Stores data in the OpenSocial gadget */
    function addInput(){
        // Getting the state
        var state = wave.getState();

        // Retrieves topics from storage.
        var jsonString = state.get('topics','[]');

        // Converts JSON to an array of topics
        var topics = toObject(jsonString);

        // Push textbox value into the array and set the textbox to blank
        topics.push(document.getElementById('textBox').value);
        document.getElementById('textBox').value = '';

        // Create an array for the topic and add it to the "master" array.
        var votes = toObject(state.get('votes','[]'));
        votes.push(new Array());

        // Submit everything to storage
        state.submitDelta({'topics' : toJSON(topics), 'votes' : toJSON(votes)});
    }
    /**
    * STEP 3 (Rendering topics)
    * - Get state
    * - Retrieve topics
    * - Add topics to the canvas
    * - Create "Add topic" button to the footer
    * - Adjust window size dynamically
    */
    // Renders the gadget
    function renderInfo() {
        /** Get state */
        if (!wave.getState()) {
            return;
        }
        var state = wave.getState();

        /** Retrieve topics */
        var topics = toObject(state.get('topics','[]'));
        var votes = toObject(state.get('votes','[]'));

        /** Add topics to the canvas */
        var html = "";
        for (var i = 0; i < topics.length; i++){
            var id = "topic"+i;
            html += '<div class="topic"><h4> ' + topics[i] + '</h4></div>';
        }
        document.getElementById('body').innerHTML = html;

        /** Create "Add topic" button to the footer */
        html += '<input type="text" id="textBox" value=""/><button id="addInput"
onclick="addInput()">Add Topic</button>';
        document.getElementById('footer').innerHTML = html;

        /** Adjust window size dynamically */
        gadgets.window.adjustHeight();
    }
}

```

adding\_topics.js is externally linked to adding\_topics.html as shown below:

```
<script type="text/javascript" src="adding_topics.js"></script>
```

adding\_topics.html is externally linked to adding\_topics.xml as shown below:

```
<Content type="html" href="adding_topics.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of adding\_topics.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_04\\_Adding\\_topics/adding\\_topics.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_04_Adding_topics/adding_topics.xml) ➡

Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> ➡

## Next

Proceed to [Lesson 5 - Voting \[page 98\]](#).

## 4.1.5.1.5 Lesson 5 - Voting

This fifth lesson of the tutorial expands on the previous tutorial by adding a voting mechanism that uses a "+" button for voting. Voters will show up as user thumbnails under the topic that they have voted for, and the voter's name will display when a user hovers over a thumbnail.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> ➡

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open voting.xml. This file is an exact copy of adding\_topics.xml that has had the title renamed to "Tutorial\_05\_Voting" as shown below:

```
<!--
* STEP 1
* - Rename our application
-->
<ModulePrefs title="Tutorial_05_Voting" height="250">
  <Require feature="wave"/>
  <Require feature="dynamic-height"/>
</ModulePrefs>
```

You will also notice other changes in voting.xml. Ignore them for now, we will discuss them further into the tutorial.

Open voting.css. It is an exact copy of adding\_topics.css and contains all of the CSS styling for our gadget as shown below:

```
/* Add CSS styling to set properties for app, header, body and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

voting.css is externally linked to voting.html as shown below:

```
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="voting.css">
```

Open voting.js. It contains all of the Javascript used in our gadget. Take note of Steps 2 and 3 in the source code and all content marked with /\*\* and \*/ shown below:

```
/**
* STEP 2 (Implement a voting button)
* - Get state and the viewer ID
* - Retrieve votes from storage and convert JSON to an array of votes
* - Push the viewer to the array of votes for the topic specified in the
function parameter
* - Convert the array to JSON and submit it to storage
*/
function vote(topicId){
  /** Get state and the viewer ID */
  var state = wave.getState();
  var viewerId = wave.getViewer().getId();

  /** Retrieve votes from storage and convert JSON to an array of votes */
  var votes = toObject(state.get('votes'));

  /** Push the viewer to the array of votes for the topic specified in the
function parameter */
  votes[topicId].push(viewerId);

  /** Convert the array to JSON and submit it to storage */
  state.submitDelta({'votes' : toJSON(votes)});
```

```

}
/**
 * STEP 3
 * - Add thumbnail rendering
 * - Add button rendering
 */
// Renders the gadget
function renderInfo() {
    // Get state
    if (!wave.getState()) {
        return;
    }
    var state = wave.getState();

    // Retrieve topics
    var topics = toObject(state.get('topics', '[]'));
    var votes = toObject(state.get('votes', '[]'));

    // Add topics to the canvas
    var html = "";
    for (var i = 0; i < topics.length; i++){
        var id = "topic"+i;
        html += '<div class="topic"><h4> ' + topics[i] + '</h4></div>';

        /** Add thumbnail rendering */
        for (var j = 0; !(typeof votes[i] === 'undefined') && j <
votes[i].length; j++){
            var voter = wave.getParticipantById(votes[i][j]);
            var thumbnail = voter.getThumbnailUrl();
            var name = voter.getDisplayName();
            html += '</button>';
    }
    document.getElementById('body').innerHTML = html;

    // Create "Add topic" button to the footer
    html = '<input type="text" id="textBox" value=""><button id="addInput"
onclick="addInput()">Add Topic</button>';
    document.getElementById('footer').innerHTML = html;

    // Adjust window size dynamically
    gadgets.window.adjustHeight();
}

```

voting.js is externally linked as shown below in voting.html as shown below:

```
<script type="text/javascript" src="voting.js"></script>
```

voting.html is externally linked to voting.xml as shown below:

```
<Content type="html" href="voting.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of voting.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_05\\_Voting/voting.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_05_Voting/voting.xml) 📄



Congratulations! Our gadget should be up and running!

### Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

### Next

Proceed to [Lesson 6 - Adding count \[page 101\]](#).

## 4.1.5.1.6 Lesson 6 - Adding count

This sixth lesson of the tutorial expands on the previous tutorial by adding the voting count of participants and group members to our gadget.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> .

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open `adding_count.xml`. This file is an exact copy of `voting.xml` that has had the title renamed to "Tutorial\_06\_Adding\_count" as shown below:

```
<!--
* STEP 1
* - Rename our application
-->
<ModulePrefs title="Tutorial_06_Adding_count" height="250">
  <Require feature="wave"/>
  <Require feature="dynamic-height"/>
</ModulePrefs>
```

You will also notice other changes in `adding_count.xml`. Ignore them for now, we will discuss them further into the tutorial.

Open `adding_count.css`. It is an exact copy of `voting.css` and contains all of the CSS styling for our gadget as shown below:

```
/* Add CSS styling to set properties for app, header, body and footer elements.
*/
```

```
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

adding\_count.css is externally linked to adding\_count.html as shown below:

```
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="adding_count.css">
```

Open adding\_count.js. It contains all of the Javascript used in our gadget. Take note of Step 2 in the source code and all content marked with `/**` and `*/` shown below:

```
/**
 * STEP 2
 * - Get the total number of votes
 * - Get the total number of wave participants
 * - Render these metrics
 */
// Renders the gadget
function renderInfo() {
    // Get state
    if (!wave.getState()) {
        return;
    }
    var state = wave.getState();

    // Retrieve topics
    var topics = toObject(state.get('topics','[]'));
    var votes = toObject(state.get('votes','[]'));

    /** Get the total number of votes */
    var voterIds = [];

    // Add topics to the canvas
    var html = "";
    for (var i = 0; i < topics.length; i++){
        var id = "topic"+i;
        html += '<div class="topic"><h4> ' + topics[i] + '</h4></div>';

        // Add thumbnail rendering
        for (var j = 0; !(typeof votes[i] === 'undefined') && j <
votes[i].length; j++){

            /** Get the total number of votes */
            if (voterIds.indexOf(votes[i][j]) == -1){
                voterIds.push(votes[i][j]);
            }

            var voter = wave.getParticipantById(votes[i][j]);
            var thumbnail = voter.getThumbnailUrl();
            var name = voter.getDisplayName();
            html += '';
        }

        // Add button rendering
        html += '<button id="voteButton" onclick="vote('+i+')"></button>';
    }
}
```

```

    }
    document.getElementById('body').innerHTML = html;

    /** Get the total number of votes */
    var votersCount = voterIds.length;

    /** Get the total number of wave participants */
    var participantsCount = wave.getParticipants().length;

    /** Render these metrics */
    html="";
    if(topics.length > 1){
        html+='<h6>'+votersCount+' out of '+participantsCount+' voted. </h6>';
    }
    html+='<input type="text" id="textBox" value=""/><button id="addInput"
onclick="addInput()" ">Add</button>';

    // Create "Add topic" button in the footer
    document.getElementById('footer').innerHTML = html;

    // Adjust window size dynamically
    gadgets.window.adjustHeight();
}

```

adding\_count.js is externally linked as shown below in adding\_count.html as shown below:

```
<script type="text/javascript" src="adding_count.js"></script>
```

adding\_count.html is externally linked to adding\_count.xml as shown below:

```
<Content type="html" href="adding_count.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of adding\_count.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_06\\_Adding\\_count/adding\\_count.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_06_Adding_count/adding_count.xml) ➡

Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> ➡

## Next

Proceed to [Lesson 7 - Adding a duplicates pop-up message \[page 104\]](#).

### 4.1.5.1.7 Lesson 7 - Adding a duplicates pop-up message

This seventh and last lesson of the tutorial expands on the previous tutorial by introducing the use of a `gadgets.Minimessage` function to provide error information feedback to the user. This will correct an issue in the previous gadget where a user can currently vote for the same topic more than once.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> 📄

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

Once you have downloaded the source code, open `adding_a_duplicates_pop-up_message.xml`. This file is an exact copy of `adding_count.xml` that has had the title renamed to "Tutorial\_07\_Adding\_a\_duplicates\_pop-up\_message" and added `<Require feature="minimessage" />` in `<ModulePrefs>` as shown below:

```
<!--
* STEP 1
* - Rename our application
* - Add "minimessage" as a required feature to ModulePrefs
-->
<ModulePrefs title="Tutorial_07_Adding_a_duplicates_pop-up_message" height="250">
  <Require feature="wave"/>
  <Require feature="dynamic-height"/>
  <Require feature="minimessage"/>
</ModulePrefs>
```

You will also notice other changes in `adding_a_duplicates_pop-up_message.xml`. Ignore them for now, we will discuss them further into the tutorial.

Open `adding_a_duplicates_pop-up_message.css`. It is an exact copy of `adding_count.css` and contains all of the CSS styling for our gadget as shown below:

```
/* Add CSS styling to set properties for app, header, body and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
```

adding\_a\_duplicates\_pop-up\_message.css is externally linked as shown below in adding\_a\_duplicates\_pop-up\_message.html as shown below:

```
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="adding_a_duplicates_pop-
up_message.css">
```

Open adding\_a\_duplicates\_pop-up\_message.js. It contains all of the Javascript used in our gadget. Take note of Step 2 in the source code and all content marked with `/**` and `*/` shown below:

```
/**
 * STEP 2
 * - Check if the user has already voted for the topic, in which case an error
will be thrown
 */
function vote(topicId){
    // Get state and the viewer ID
    var state = wave.getState();
    var viewerId = wave.getViewer().getId();

    // Retrieve votes from storage and convert JSON to an array of votes
    var votes = toObject(state.get('votes'));

    /** Check if the user has already voted for the topic, in which case an
error will be thrown */
    if(votes[topicId].indexOf(viewerId) == -1){
        // Push the viewer to the array of votes for the topic specified in the
function parameter
        votes[topicId].push(viewerId);

        // Convert the array to JSON and submit it to storage
        state.submitDelta({'votes' : toJSON(votes)});
    }else{
        var prefs = new gadgets.Prefs();
        var message = new gadgets.MinMessage(prefs.getModuleId());
        message.createDismissibleMessage("Cannot vote for the same topic more
than once!");
        gadgets.window.adjustHeight();
    }
}
```

adding\_a\_duplicates\_pop-up\_message.js is externally linked to adding\_a\_duplicates\_pop-up\_message.html as shown below:

```
<script type="text/javascript" src="adding_a_duplicates_pop-up_message.js"></
script>
```

adding\_a\_duplicates\_pop-up\_message.html is externally linked to adding\_a\_duplicates\_pop-up\_message.xml as shown below:

```
<Content type="html" href="adding_a_duplicates_pop-up_message.html"/>
```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of adding\_a\_duplicates\_pop-up\_message.xml with your SAP Jam live service, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

- [https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_07\\_Adding\\_a\\_duplicates\\_pop-up\\_message/adding\\_a\\_duplicates\\_pop-up\\_message.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_07_Adding_a_duplicates_pop-up_message/adding_a_duplicates_pop-up_message.xml) 📄

Congratulations! You have completed the tutorial and our gadget should be up and running!!

### Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> 📄 .

## Next

If you would like to review the full code of the tutorial, see [Complete Tutorial Code Example \[page 106\]](#).

## 4.1.5.1.8 Complete Tutorial Code Example

This final page of the tutorial presents the full code examples of the gadget developed in the preceding lessons of the tutorial.

In this tutorial we will only be working with source code downloaded from the official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> 📄

The code snippets in the tutorial are not for copy/paste operations for building gadgets. Please use our source code to copy/paste from and as templates for your own gadgets.

Each lesson creates a fully working gadget that you can run either from your own web server or from ours.

The gadget is composed of the following files:

- complete\_tutorial\_code\_example.xml
- complete\_tutorial\_code\_example.html
- complete\_tutorial\_code\_example.js
- complete\_tutorial\_code\_example.css

complete\_tutorial\_code\_example.xml is shown below:

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Tutorial_08_Complete_Tutorial_Code_Example (XML, HTML, CSS, JavaScript) -->
<Module specificationVersion='2'>
  <ModulePrefs title="Tutorial_08_Complete_Tutorial_Code_Example" height="250">
```

```

        <Require feature="wave"/>
        <Require feature="dynamic-height"/>
        <Require feature="minimessage"/>
    </ModulePrefs>
    <Content type="html" href="complete_tutorial_code_example.html"/>
</Module >

```

complete\_tutorial\_code\_example.html is shown below:

```

<!DOCTYPE html>
<!-- Add interactive functionality via Javascript -->
<script type="text/javascript" src="complete_tutorial_code_example.js"></script>
<!-- Add some basic HTML structure to our gadget-->
<div id="app">
    <div id="header"><h2>OpenSocial Tutorial Tool</h2></div>
    <div id="body"></div>
    <div id="footer"></div>
</div>
<!-- Add CSS styling to set properties for app, header, body and footer
elements. -->
<link rel="stylesheet" type="text/css" href="complete_tutorial_code_example.css">

```

complete\_tutorial\_code\_example.js is shown below:

```

/** Add interactive functionality via Javascript */
// Adds voting functionality
function vote(topicId){
    // Get state and the viewer ID
    var state = wave.getState();
    var viewerId = wave.getViewer().getId();

    // Retrieve votes from storage and convert JSON to an array of votes
    var votes = toObject(state.get('votes'));

    // Check if the user has already voted for the topic, in which case an error
    will be thrown
    if(votes[topicId].indexOf(viewerId) == -1){
        // Push the viewer to the array of votes for the topic specified in the
        function parameter
        votes[topicId].push(viewerId);

        // Convert the array to JSON and submit it to storage
        state.submitDelta({'votes' : toJSON(votes)});
    }else{
        var prefs = new gadgets.Prefs();
        var message = new gadgets.Minimessage(prefs.getModuleId());
        message.createDismissibleMessage("Cannot vote for the same topic more
        than once!");
        gadgets.window.adjustHeight();
    }
}
// Encode object as JSON string
function toJSON(obj) {
    return gadgets.json.stringify(obj);
}
// Decode JSON string into an object
function toObject(str) {
    return gadgets.json.parse(str);
}
// Stores data in the OpenSocial gadget
function addInput(){
    // Getting the state
    var state = wave.getState();

    // Retrieves topics from storage.
    var jsonString = state.get('topics','[]');

```

```

// Converts JSON to an array of topics
var topics = toObject(jsonString);

// Push textbox value into the array and set the textbox to blank
topics.push(document.getElementById('textBox').value);
document.getElementById('textBox').value = '';

// Create an array for the topic and add it to the "master" array.
var votes = toObject(state.get('votes','[]'));
votes.push(new Array());

// Submit everything to storage
state.submitDelta({'topics' : toJSON(topics), 'votes' : toJSON(votes)});
}
// Renders the gadget
function renderInfo() {
    // Get state
    if (!wave.getState()) {
        return;
    }
    var state = wave.getState();

    // Retrieve topics
    var topics = toObject(state.get('topics','[]'));
    var votes = toObject(state.get('votes','[]'));

    // Get the total number of votes
    var voterIds = [];

    // Add topics to the canvas
    var html = "";
    for (var i = 0; i < topics.length; i++){
        var id = "topic"+i;
        html += '<div class="topic"><h4> ' + topics[i] + '</h4></div>';

        // Add thumbnail rendering
        for (var j = 0; !(typeof votes[i] === 'undefined') && j <
votes[i].length; j++){

            // Get the total number of votes
            if (voterIds.indexOf(votes[i][j]) == -1){
                voterIds.push(votes[i][j]);
            }

            var voter = wave.getParticipantById(votes[i][j]);
            var thumbnail = voter.getThumbnailUrl();
            var name = voter.getDisplayName();
            html += '';
        }

        // Add button rendering
        html += '<button id="voteButton" onclick="vote('+i+')"></button>';
    }
    document.getElementById('body').innerHTML = html;

    // Get the total number of votes
    var votersCount = voterIds.length;

    // Get the total number of wave participants
    var participantsCount = wave.getParticipants().length;

    // Render these metrics
    html="";
    if(topics.length > 1){
        html+='\<h6>'+votersCount+' out of '+participantsCount+' voted. </h6>';
    }
}

```



```

    html+='\<input type="text" id="textBox" value=""/>\<button id="addInput"
onclick="addInput()">Add\</button>';

    // Create "Add topic" button in the footer
    document.getElementById('footer').innerHTML = html;

    // Adjust window size dynamically
    gadgets.window.adjustHeight();
}
// Initializes gadget, sets callbacks
function init() {
    if (wave && wave.isInWaveContainer()) {
        // Loads the gadget's initial state and the subsequent changes to it
        wave.setStateCallback(renderInfo);

        // Loads participants and any changes to them
        wave.setParticipantCallback(renderInfo);
    }
}
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);

```

complete\_tutorial\_code\_example.css is shown below:

```

/* Add CSS styling to set properties for app, header, body and footer elements.
*/
#app {line-height: 20px;border: 1px solid #EEEEEE;}
#header {padding-top: 10px;
padding-bottom: 10px;
margin-left: auto;
margin-right: auto;
width: 40%;
color: #5c5c5c;}
#body {color: #5c5c5c;padding-left: 30px;}
#footer {padding: 30px;}
#voteButton {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}
#thumbnail {width: 40px;height: 40px;padding: 2px;vertical-align:middle;}

```

## RUN OUR GADGET!

Upload the source files of our gadget to your publicly accessible server, register the URL of complete\_tutorial\_code\_example.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial\\_08\\_Complete\\_Tutorial\\_Code\\_Example/complete\\_tutorial\\_code\\_example.xml](https://sapjamsamplecode.github.io/OpenSocial/Gadget/Tutorial/Tutorial_08_Complete_Tutorial_Code_Example/complete_tutorial_code_example.xml) ➡

Congratulations! You have completed the tutorial and our gadget should be up and running!!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)

- <https://developers.google.com/gadgets/docs/ui>  .

## 4.1.5.2 Intermediate: API Gadget Tutorials

This section provides tutorials that demonstrate all of the features of an OpenSocial or Apache Wave API endpoint in a simple SAP Jam Collaboration OpenSocial gadget.



### 4.1.5.2.1 Context Gadgets

This section provides tutorials that demonstrate all of the features of the OpenSocial Context API. A simple SAP Jam Collaboration OpenSocial gadget is used to demonstrate each endpoint.

#### 4.1.5.2.1.1 Content Instance Context Tutorial

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to display the context of a content instance.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> .
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/context/contextcontentgadget )
3. Download the minified jQuery 2.x library from <http://jquery.com/download/> .
4. Copy the minified jQuery 2.x library into the gadget's folder ( ../OpenSocial/Gadget/context/contextcontentgadget )

Once you have completed these steps, open **SAPJAM\_context.html**, adjust the jQuery src filename to match the filename of the minified jQuery 2.x library in your gadget folder and save.

```
<html>
<head>
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <!--
    * STEP 1
    * - Adjust the jQuery src filename to match the jQuery 2.x version in your
    gadget folder
  -->
  <script type="text/javascript" src="jquery-2.2.3.min.js"></script>
  <script type="text/javascript" src="SAPJAM_context.js"></script>
</head>
<body>
</body>
</html>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how the context of the Content Instance is found using **gadgets.sapjam.context.get**.

Open **SAPJAM\_context.js** and read all of the in-line comments in the code to learn how **gadgets.sapjam.context.get** works within this gadget.

```
function make_SAPJAM_context_Call() {
    /*
        Display all of the properties for "gadgets.sapjam.context.get" in a HTML
        page.
    */

    gadgets.sapjam.context.get(function(data) {
        console.log(JSON.stringify(data, null, 4));
        /* Begin HTML page */
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
        "<p>-----</p>";
        osapiOutput += "<p><h2>gadgets.sapjam.context.get</h2></p>";
        osapiOutput += "<ul>";
        /* Adds all the properties of "gadgets.sapjam.context.get" with HTML
        formatting to a string (osapiOutput). */
        osapiOutput += "<li>data.<b>context</b> = " + data.context + "</li>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>name</b> = " + data.name + "</li>";
        osapiOutput += "<li>data.<b>readOnly</b> = " + data.readOnly + "</li>";

        /* Adds group context object properties to the string (osapiOutput) if
        the gadget has a content or group context. */
        osapiOutput += "<li>Group Context Object:<ul>";
        if (data.context != "preview") {
            osapiOutput += "<li>data.<b>group.id</b> = " + data.group.id + "</li>";
            osapiOutput += "<li>data.<b>group.name</b> = " + data.group.name +
            "</li>";
        }
        else {
            osapiOutput += "<li>Group Context Object not available. This gadget
            must be added to a group to have a Group Context Object.</li>";
        }
        osapiOutput += "</ul></li>";
        /* End HTML page */
        osapiOutput += "<li>Open your browser console to view the raw JSON
        object.</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
        "<p>-----</p>";
        /* Displays the string (osapiOutput). */
        $("body").append(osapiOutput);
    });
}
// Initializes gadget
function init() {
    make_SAPJAM_context_Call();
}
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);
```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **SAPJAM\_context.xml** with SAP Jam Collaboration as a **Content** gadget, create a group, add the gadget to that group, and run it.

Congratulations! This gadget should be up and running!

### Troubleshooting

If you are having any problems registering this gadget with SAP Jam Collaboration, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### Reference



To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui>  .

## 4.1.5.2.1.2 Group Gadget Instance Context Tutorial

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to display the context of a group gadget instance.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> 
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/context/contextgroupgadget )
3. Download the minified jQuery 2.x library from <http://jquery.com/download/> 
4. Copy the minified jQuery 2.x library into the gadget's folder ( ../OpenSocial/Gadget/context/contextgroupgadget )

Once you have completed these steps, open **SAPJAM\_context.html**, adjust the jQuery src filename to match the filename of the minified jQuery 2.x library in your gadget folder and save.

```
<html>
  <head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!--
    * STEP 1
    * - Adjust the jQuery src filename to match the jQuery 2.x version in your
    gadget folder
    -->
    <script type="text/javascript" src="jquery-2.2.3.min.js"></script>
    <script type="text/javascript" src="SAPJAM_context.js"></script>
  </head>
  <body>
  </body>
</html>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how the context of the Content Instance is found using **gadgets.sapjam.context.get**.

Open **SAPJAM\_context.js** and read all of the in-line comments in the code to learn how **gadgets.sapjam.context.get** works within this gadget.

```
function make_SAPJAM_context_Call() {
  /*
    Display all of the properties for "gadgets.sapjam.context.get" in a HTML
    page.
```

```

    */
    gadgets.sapjam.context.get(function(data) {
        console.log(JSON.stringify(data, null, 4));
        /* Begin HTML page */
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
        "<p>-----</p>";
        osapiOutput += "<p><h2>gadgets.sapjam.context.get</h2></p>";
        osapiOutput += "<ul>";
        /* Adds all the properties of "gadgets.sapjam.context.get" with HTML
        formatting to a string (osapiOutput). */
        osapiOutput += "<li>data.<b>context</b> = " + data.context + "</li>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>name</b> = " + data.name + "</li>";
        osapiOutput += "<li>data.<b>readOnly</b> = " + data.readOnly + "</li>";

        /* Adds group context object properties to the string (osapiOutput) if
        the gadget has a content or group context. */
        osapiOutput += "<li>Group Context Object:<ul>";
        if (data.context != "preview") {
            osapiOutput += "<li>data.<b>group.id</b> = " + data.group.id + "</
            li>";
            osapiOutput += "<li>data.<b>group.name</b> = " + data.group.name +
            "</li>";
        }
        else {
            osapiOutput += "<li>Group Context Object not available. This gadget
            must be added to a group to have a Group Context Object.</li>";
        }
        osapiOutput += "</ul></li>";
        /* End HTML page */
        osapiOutput += "<li>Open your browser console to view the raw JSON
        object.</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
        "<p>-----</p>";
        /* Displays the string (osapiOutput). */
        $("body").append(osapiOutput);
    });
}
// Initializes gadget
function init() {
    make_SAPJAM_context_Call();
}
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);

```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **SAPJAM\_context.xml** with SAP Jam Collaboration as a **Group** gadget and use the following steps to create a **PrimaryGadgetObject** off a group using our OData API:

1. Register [hello\\_world.xml](#) with SAP Jam as a group gadget.
2. Copy the **GroupGadget Id** from the group gadget's URL ([https://<jam#>.sapjam.com/open\\_social\\_gadgets/GROUPGADGET\\_ID](https://<jam#>.sapjam.com/open_social_gadgets/GROUPGADGET_ID)).
3. Download [create\\_primary\\_gadget\\_object.json](#). This file will create your PrimaryGadgetObject.
4. Open this file and replace 'ENTER\_YOUR\_GROUPGADGET\_ID' with the **GroupGadget Id**.

5. Create a group in SAP Jam. A new group appears.
6. Copy the **Group Id** from the URL ([https://<jam#>.sapjam.com/groups/about\\_page/GROUP\\_ID](https://<jam#>.sapjam.com/groups/about_page/GROUP_ID)).
7. Run the following CURL command using the copied **Group Id** (`/api/v1/OData/Groups('YOUR_GROUP_ID')/PrimaryGadgetObject`) and **create\_primary\_gadget\_object.json** (`-d@C:/YOUR_PRIMARY_GADGET_PATH/create_primary_gadget_object.json`).

Congratulations! This gadget should be up and running!

### Troubleshooting

If you are having any problems registering this gadget with SAP Jam, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### Reference



To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui>  .

## 4.1.5.2.1.3 Profile Gadget Instance Context Tutorial

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to display the context of a profile gadget instance.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> 
2. Extract the zip file and navigate to the gadget folder ( `../OpenSocial/Gadget/context/contextprofilegadget` )
3. Download the minified jQuery 2.x library from <http://jquery.com/download/> 
4. Copy the minified jQuery 2.x library into the gadget's folder ( `../OpenSocial/Gadget/context/contextprofilegadget` )

Once you have completed these steps, open **SAPJAM\_context.html**, adjust the jQuery src filename to match the filename of the minified jQuery 2.x library in your gadget folder and save.

```
<html>
  <head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!--
      * STEP 1
      * - Adjust the jQuery src filename to match the jQuery 2.x version in your
      gadget folder
    -->
    <script type="text/javascript" src="jquery-2.2.3.min.js"></script>
    <script type="text/javascript" src="SAPJAM_context.js"></script>
  </head>
  <body>
  </body>
</html>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how the context of the Content Instance is found using **gadgets.sapjam.context.get**.

Open **SAPJAM\_context.js** and read all of the in-line comments in the code to learn how **gadgets.sapjam.context.get** works within this gadget.

```
function make_SAPJAM_context_Call(){
    gadgets.sapjam.context.get(function(data) {
        console.log(JSON.stringify(data, null, 4));
        /* Begin HTML page */
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
        "<p>-----</p>";
        osapiOutput += "<p><h2>gadgets.sapjam.context.get</h2></p>";
        osapiOutput += "<ul>";
        /* Adds all the properties of "gadgets.sapjam.context.get" with HTML
        formatting to a string (osapiOutput). */
        osapiOutput += "<li>data.<b>context</b> = " + data.context + "</li>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>name</b> = " + data.name + "</li>";
        osapiOutput += "<li>data.<b>readOnly</b> = " + data.readOnly + "</li>";
        /* End HTML page */
        osapiOutput += "<li>Open your browser console to view the raw JSON
        object.</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
        "<p>-----</p>";
        osapiOutput += "<p></p>";
        /* Displays the string (osapiOutput). */
        $("body").append(osapiOutput);
    });
}
// Initializes gadget
function init() {
    make_SAPJAM_context_Call();
}
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);
```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **SAPJAM\_context.xml** with SAP Jam Collaboration as a **Profile** gadget, open your profile, click **Show Additional Info** and scroll down to it.

Congratulations! This gadget should be up and running!

## Troubleshooting

If you are having any problems registering this gadget with SAP Jam Collaboration, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui>  .

## 4.1.5.2.1.4 Statusbar Gadget Instance Context Tutorial

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to display the context of a statusbar gadget instance.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> 📄
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/context/contextstatusbargadget )
3. Download the minified jQuery 2.x library from <http://jquery.com/download/> 📄
4. Copy the minified jQuery 2.x library into the gadget's folder ( ../OpenSocial/Gadget/context/contextstatusbargadget )

Once you have completed these steps, open **SAPJAM\_context.html**, adjust the jQuery src filename to match the filename of the minified jQuery 2.x library in your gadget folder and save.

```
<html>
  <head>
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <!--
    * STEP 1
    * - Adjust the jQuery src filename to match the jQuery 2.x version in your
    gadget folder
    -->
    <script type="text/javascript" src="jquery-2.2.3.min.js"></script>
    <script type="text/javascript" src="SAPJAM_context.js"></script>
  </head>
  <body>
</body>
</html>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how the context of the Content Instance is found using **gadgets.sapjam.context.get**.

Open **SAPJAM\_context.js** and read all of the in-line comments in the code to learn how **gadgets.sapjam.context.get** works within this gadget.

```
function make_SAPJAM_context_Call(){
  gadgets.sapjam.context.get(function(data) {
    console.log(JSON.stringify(data, null, 4));
    /* Begin HTML page */
    var osapiOutput = "";
    osapiOutput += "gadgets.sapjam.context.get --> ";
    /* Adds all the properties of "gadgets.sapjam.context.get" with HTML
    formatting to a string (osapiOutput). */
    osapiOutput += "context = " + data.context;
    osapiOutput += ", id = " + data.id;
    osapiOutput += ", name = " + data.name;
    osapiOutput += ", readOnly = " + data.readOnly;
    /* End HTML page */
    osapiOutput += ". Open your browser console to view the raw JSON
    object.";
    /* Appends the string (osapiOutput) to the body of the HTML page. */
    $("body").append(osapiOutput);
    /* Demonstrates every gadgets.sapjam.statusbar interaction */
    setTimeout(showStatusBar, 1000);
    setTimeout(clearBadgeText, 4000);
    setTimeout(hideStatusBar, 7000);
    setTimeout(clearBadgeText, 10000);
    setTimeout(clickToExpand, 13000);
    setTimeout(clearHighlight, 15000);
```



```

    });
}
function showStatusBar(){
    /* Highlights the gadget in blue and makes it blink 3 times */
    gadgets.sapjam.statusbar.highlight();
    /* Expands the gadget and shows the contents of the body element within it */
    gadgets.sapjam.statusbar.show();
    /* Adds a red badge with text to the gadget */
    gadgets.sapjam.statusbar.setBadgeText("Statusbar Expanded");
}
function clearBadgeText(){
    gadgets.sapjam.statusbar.clearBadgeText();
    gadgets.sapjam.statusbar.clearHighlight();
}
function hideStatusBar(){
    gadgets.sapjam.statusbar.hide();
    gadgets.sapjam.statusbar.highlight();
    gadgets.sapjam.statusbar.setBadgeText("Statusbar collapsed");
}
function clickToExpand(){
    gadgets.sapjam.statusbar.highlight();
    gadgets.sapjam.statusbar.setBadgeText("Click here");
}
function clearHighlight(){
    gadgets.sapjam.statusbar.clearHighlight();
}
// Initializes gadget
function init() {
    make_SAPJAM_context_Call();
}
// Initializes gadget after receiving a notification that the page is loaded and
// the DOM is ready.
gadgets.util.registerOnLoadHandler(init);

```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **SAPJAM\_context.xml** with SAP Jam Collaboration as a **Status Bar** gadget and click **Home**. The gadget will appear in the statusbar at the bottom of the browser window.

**Note:** **Status Bar** gadgets never activate on **Gadget Configuration** pages.

Congratulations! This gadget should be up and running!

## Troubleshooting

If you are having any problems registering this gadget with SAP Jam Collaboration, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

## 4.1.5.3 Gadget Application Tutorials

This section provides tutorials that demonstrate how to use OpenSocial and Apache Wave APIs to create feature rich SAP Jam Collaboration OpenSocial gadget applications.

### 4.1.5.3.1 To Do Gadget

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to add/remove todo items as the user navigates through Jam and displays attributes of the following OpenSocial features: gadgets.sapjam.context, dynamic-height, dynamic-width, open-views, i18N, prefs, osapi.people, and osapi.appdata.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> ↗
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/gadgetapplications/todogadgetsimple )
3. Download the minified jQuery 1.x library from <http://jquery.com/download/> ↗
4. Copy the minified jQuery 1.x library into the gadget's jQuery folder ( ../OpenSocial/Gadget/gadgetapplications/todogadgetsimple/vendor/jquery )
5. Download the bootstrap library from <http://getbootstrap.com/getting-started/#download> ↗
6. Extract the zip file and navigate to the css folder ( ../bootstrap-3.x-dist/css )
7. Copy the minified Bootstrap 3.x css file ( bootstrap.min.css ) into the gadget's Bootstrap folder ( ../OpenSocial/Gadget/gadgetapplications/todogadgetsimple/vendor/bootstrap )

Once you have completed these steps navigate to the gadget source folder ( ../todogadget ), open **spec.xml**, adjust the jQuery src filename in both locations ( Step 1, Step 2 ) to match the filename of the minified jQuery 1.x library in your gadget's vendor folder and save.

```
<Content type="html" view="home"><![CDATA[
  <!DOCTYPE html>
  <html>
    <head>
      <!--
      * STEP 1
      * - Adjust the jQuery src filename to match the jQuery 1.x version
in your gadget's vendor/jquery folder
      * - Adjust the Bootstrap 3.x css filename to match the Bootstrap 3.x
version in your gadget's vendor/bootstrap folder
      -->
      <script src="../vendor/jquery/jquery-1.11.1.min.js"></script>
      <link rel="stylesheet" type="text/css" href="../vendor/bootstrap/
bootstrap.min.css"/>
      <link rel="stylesheet" type="text/css" href="todo_list.css"/>
    </head>
```

```
<Content type="html" view="popup-view"><![CDATA[
  <!DOCTYPE html>
  <html>
    <head>
      <!--
      * STEP 2
      * - Adjust the jQuery src filename to match the jQuery 1.x version
in your gadget's vendor/jquery folder
```

```

        * - Adjust the Bootstrap 3.x css filename to match the Bootstrap 3.x
        version in your gadget's vendor/bootstrap folder
        -->
        <script src="../../vendor/jquery/jquery-1.11.1.min.js"></script>
        <link rel="stylesheet" type="text/css" href="../../vendor/bootstrap/
bootstrap.min.css"/>
        <link rel="stylesheet" type="text/css" href="todo_list.css"/>
    </head>

```

The gadget is now ready to use! Before we test the gadget, let's explore more about how this gadget works.

Open **todo\_list.js** and read all of the in-line comments in the code to learn more about how `osapi.people.getOwner`, `osapi.appdata.get`, `gadgets.views.openGadget` and `osapi.appdata.update` are used in the gadget.

```

var todoController = function() {
    return {
        init: function() {

            /** Keep track of all the existing 'todo items' that have not been
            removed */
            var existingTodos = [];
            /**
            * Retrieve and display the gadget owner's 'todo items'.
            * - osapi.people.getOwner retrieves the owner of the gadget.
            * - osapi.appdata.get retrieves the gadget owner's 'todo items' by
            specifically calling for the ['todos'] keys.
            * - osapi.appdata.get retrieves the gadget owner's 'todo items' from
            the ['todos'] keys.
            */
            function getExistingTodos() {
                // Retrieves the owner of the gadget
                osapi.people.getOwner().execute(function(ownerData) {
                    var ownerId = ownerData.id;
                    // Retrieves the gadget owner's 'todo items'
                    osapi.appdata.get({
                        userId: ownerId,
                        keys: ['todos']
                    }).execute(function(todos) {
                        existingTodos = (todos[ownerId] !== undefined &&
                        todos[ownerId].todos !== undefined) ? todos[ownerId].todos : [];

                        // Display the gadget owner's 'todo items'
                        displayTodos();
                    });
                });
            }

            /**
            * Display each complete/uncomplete 'todo item' in a list of clickable
            buttons.
            * - Clicking a button when its button text displays 'Mark as Completed'
            will:
            *   - Mark the related 'todo item' as 'Completed'
            *   - Change the button text to 'Completed'
            *   - Change the button color to green
            *   - Change the button text color to white
            * - Clicking a button when its button text displays 'Completed' will
            not do anything.
            */
            function displayTodos() {
                // Remove all items from the todo list
                clearTodos();
                if (existingTodos.length > 0) {
                    var id = 1;
                    $.each(existingTodos, function(i, val) {

```

```

        var htmlString = null;
        if (val.completed) {
            htmlString = getCompletedItem(val.name, id);
        } else {
            htmlString = getUncompletedItem(val.name, id);
        }
        $('#todos').append(htmlString);
        $('#'+ id).click(function() {
            markTodoAsCompleted(val);
        });
        id++;
    });
}
// Adjust the height of the gadget window for added or removed 'todo
items'.
    gadgets.window.adjustHeight();
}

/**
 * Return HTML for a completed 'todo item'
 */
function getCompletedItem(name, id) {
    return "<div class=\"input-group list-group-item\">" +
        name +
        "<span class=\"input-group-btn\"><button type=\"button\" id="
    \"" +
        id +
        "\"\" +
        "class=\"btn btn-success complete-task\">" +
        gadgetPrefs.getMsg("completed_item") +
        "</button></span></div>";
}

/**
 * Return HTML for an uncompleted 'todo item'
 */
function getUncompletedItem(name, id) {
    return "<div class=\"input-group list-group-item\">" +
        name +
        "<span class=\"input-group-btn\"><button type=\"button\" id="
    \"" +
        id +
        "\"\" +
        "class=\"btn btn-default complete-task\">" +
        gadgetPrefs.getMsg("uncompleted_item") +
        "</button></span></div>";
}

/**
 * Remove all items from the todo list
 */
function clearTodos() {
    $('#todos').empty();
}

/**
 * Open a modal dialog with the HTML rendered for the 'popup-view' open-
view.
 * - gadgets.views.openGadget allows for different content from the
gadget definition to be rendered.
 * Refer to <Content type="html" view="popup-view"> in the spec.xml
file.
 */
function openView() {
    gadgets.views.openGadget(function(result) {
        return result ? createTodo(result) : null;
    }, function(site) {
        return null;
    });
}

```

```

        }, {
            view: 'popup-view',
            viewTarget: 'MODALDIALOG'
        });
    }

    /**
     * Creates a 'todo item' by updating the existingTodos.
     * - osapi.appdata.update updates the 'todos' key for the gadget with
the new todo added.
     * - Gets the existing 'todo items' after the update.
     */
    function createTodo(todo) {
        existingTodos.push(todo);
        // Updates the 'todos' key
        osapi.appdata.update({
            data: {
                todos: existingTodos
            }
        }).execute(function(updateData) {
            if (updateData.error) {
                window.console && console.log(updateData.error.message);
            }
        });
        // Gets the existing 'todo items'
        getExistingTodos();
    }

    /**
     * Removes 'todo items' by updating the existingTodos.
     * - osapi.appdata.update updates the 'todos' key for the gadget with
the remaining gadgets after the
     * removal of completed todos.
     * - Gets the existing 'todo items' after the update.
     */
    function removeCompletedTodos() {
        var newTodos = [];
        $.each(existingTodos, function(i, val) {
            if (!(val.completed)) {
                newTodos.push(val);
            }
        });
        // Updates the 'todos' key
        osapi.appdata.update({
            data: {
                todos: newTodos
            }
        }).execute(function(updateData) {
            if (updateData.error) {
                window.console && console.log(updateData.error.message);
            }
        });
        // Gets the existing 'todo items'
        getExistingTodos();
    }

    /**
     * Marks a todo item as completed. This marks all tasks with the same
description as completed, so
     * duplicates will be marked as complete.
     * - Gets the context of the gadget to determine whether the current
user has the appropriate
     * permissions to complete a task.
     * - osapi.appdata.update updates the 'todos' key for the gadget with
the updated 'todo items'.
     * - Gets the existing 'todo items' after the update.
     */

```

```

function markTodoAsCompleted(todo) {

    // Gets the context of the gadget
    gadgets.sapjam.context.get(function(context) {
        if (!context.readOnly) {
            $.each(existingTodos, function(i, val) {
                if (val.name === todo.name) {
                    existingTodos[i].completed = true;
                }
            });
            // Updates the 'todos' key
            osapi.appdata.update({
                data: {
                    todos: existingTodos
                }
            }).execute(function(updateData) {
                if (updateData.error) {
                    window.console && console.log(updateData.error.message);
                }
                // Gets the existing 'todo items'
                getExistingTodos();
            });
        }
    });
}

/**
 * Gets all existing 'todo items' and then sets click listeners for the
 * different buttons.
 */
function initGadget() {
    // Gets all existing 'todo items'
    getExistingTodos();
    // Sets click listeners for the different buttons
    $('#new-todo').click(function() {
        openView();
    });
    $('#remove-completed-todos').click(function() {
        removeCompletedTodos();
    });
}

/**
 * Establishes the gadget preferences feature
 */
var gadgetPrefs = new gadgets.Prefs();
/**
 * Initializes the gadget after receiving a notification that the page
 * is loaded and the DOM is ready.
 */
gadgets.util.registerOnLoadHandler(initGadget);
}
};
}();

```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **spec.xml** with SAP Jam as a **Content** gadget, create a group, add the gadget to that group, and run it.

Congratulations! This gadget should be up and running!

## Troubleshooting

If you are having any problems registering this gadget with SAP Jam, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference


To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

## 4.1.5.3.2 Simple Multi-User Sync Gadget

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget that creates a communication stream between 1 gadget instance in 1 group between multiple users. This gadget allows multiple users to create and edit comments as drafts privately before publishing these comments to a public shared stream of comments by using the following OpenSocial features: wave, dynamic-height, and osapi.activitystreams.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> 
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/gadgetapplications/commentstreamgadget/comment\_stream\_gadget )
3. Download the minified jQuery 1.x library from <http://jquery.com/download/> 
4. Copy the minified jQuery 1.x library into the gadget's jQuery folder ( ../OpenSocial/Gadget/gadgetapplications/commentstreamgadget/vendor/jquery )
5. Download the bootstrap library from <http://getbootstrap.com/getting-started/#download> 
6. Extract the zip file and navigate to the css folder ( ../bootstrap-3.x-dist/css )
7. Copy the minified Bootstrap 3.x css file ( bootstrap.min.css ) into the gadget's Bootstrap folder ( ../OpenSocial/Gadget/gadgetapplications/commentstreamgadget/vendor/bootstrap )

Once you have completed these steps navigate to the gadget source folder ( ../comment\_stream\_gadget ), open **comment\_stream.html**, adjust the jQuery src filename to match the filename of the minified jQuery 1.x library in your gadget's vendor folder and save.

```
<!DOCTYPE html>
<html>
  <head>
    <!--
      * STEP 1
      * - Adjust the jQuery src filename to match the jQuery 1.x version in your
      gadget's vendor/jquery folder
      * - Adjust the Bootstrap 3.x css filename to match the Bootstrap 3.x version
      in your gadget's vendor/bootstrap folder
    -->
    <script src="../../vendor/jquery/jquery-1.11.1.min.js"></script>
    <link rel="stylesheet" type="text/css" href="../../vendor/bootstrap/
bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css" href="comment_stream.css"/>
  </head>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how this gadget works.

Open **comment\_stream.js** and read all of the in-line comments in the code to learn more about how wave, dynamic-height, and osapi.activitystreams are used in the gadget.

```
var commentStreamController = function() {
  return {
    init: function() {

      /**
       * Clears all of the current user's drafts.
       * - Uses wave.getPrivateState().submitDelta(map) to update the private
       state object (wave) with
       * a passed in map of key-values for the current user. This user-
       specific information is
       * private and can only be accessed by that user.
       */
      function clearDrafts() {
        // Initializes the 'drafts' array for the current user in the
        private state object (wave)
        wave.getPrivateState().submitDelta({'drafts': []});

        $('#drafts').empty();
      }

      /**
       * Edits a drafted comment and remove it from the list of drafts.
       * - Uses wave.getPrivateState().submitDelta(map) to update the private
       state object (wave) with
       * a passed in map of key-values for the current user. This user-
       specific information is
       * private and can only be accessed by that user.
       */
      function editDraft(draft) {
        $('#drafts').empty();
        $('#comment-text').val(draft);
        var currentDrafts = getDrafts();
        var updatedDrafts = [];
        $.each(currentDrafts, function(i, val) {
          if (val !== draft) {
            updatedDrafts.push(val);
          }
        });

        // Updates the 'drafts' array with the current user's drafts in the
        private state object (wave)
        wave.getPrivateState().submitDelta({'drafts': updatedDrafts});
      }

      /** Returns the HTML for a comment. */
      function getCommentHtml(comment) {
        return "<div>" + comment + "</div>";
      }

      /** Returns the HTML for a draft. */
      function getDraftHtml(draft, id) {
        return "<div class=\"input-group list-group-item\">" +
          draft +
          "<span class=\"input-group-btn\"><button type=\"button\" id=" +
          id +
          "\" class=\"btn btn-default complete-task\">Edit</button></" +
          span></div>";
      }

      /**
       * Renders comments in the comment stream output box.

```



```

    * - Uses gadgets.window.adjustHeight() to resize the gadget's window
    height to fit added/removed content.
    */
    function renderComments(commentsToRender) {
        $('#output-box').empty();
        if (commentsToRender.length > 0) {
            $.each(commentsToRender, function(i, val) {
                var htmlString = getCommentHtml(val);
                $('#output-box').append(htmlString);

                // Resizes the gadget's window height to fit content
                gadgets.window.adjustHeight();
            });
        }
    }

    /**
    * Renders drafts in the drafts box.
    * - Uses gadgets.window.adjustHeight() to resize the gadget's window
    height to fit added/removed content.
    */
    function renderDrafts(draftsToRender) {
        $('#drafts').empty();
        if (draftsToRender.length > 0) {
            var id = 1;
            $.each(draftsToRender, function(i, val) {
                var htmlString = getDraftHtml(val, id);
                $('#drafts').append(htmlString);

                // Resizes the gadget's window height to fit content
                gadgets.window.adjustHeight();
                $('#' + id).click(function() {
                    editDraft(val);
                });
                id++;
            });
        }
    }

    /**
    * Gets all drafts for the current user.
    * - Uses wave.getPrivateState().get(map) to get a map of key-values
    from the private state object (wave) for the
    * current user. This user-specific information is private and can
    only be accessed by that user.
    */
    function getDrafts() {
        // Gets an array of all drafts for the current user from the private
        state object (wave)
        var draftsToRender = wave.getPrivateState().get('drafts') === null ?
        [] : wave.getPrivateState().get('drafts');
        return draftsToRender;
    }

    /**
    * Gets all comments in the shared state object (wave), which applies to
    all users.
    * - Uses wave.getState().get(map) to get a map of key-values from the
    shared state object (wave). All
    * information is public and can be accessed by all users.
    */
    function getComments() {
        // Gets an array of all user comments from the public shared object
        (wave)
        var commentsToRender = wave.getState().get('comments') === null ?
        [] : wave.getState().get('comments');
        return commentsToRender;
    }

```

```

/**
 * 1. Publishes a comment to the shared state object (wave) comment
stream.
 * - Uses wave.getState().submitDelta(map) to update the shared state
object (wave) with a
 * passed in map of key-values. All information is public and can
be accessed by all users.
 * 2. Removes any draft that was published, as it is no longer a draft.
 * - Uses wave.getPrivateState().submitDelta(map) to update the
private state object (wave) with
 * a passed in map of key-values for the current user. This user-
specific information is
 * private and can only be accessed by that user.
 * 3. Creates a new comment in the feed of the gadget's Jam group.
 * - Uses osapi.activitystreams.create({
 * activity: {
 * title: "titleText",
 * object: {
 * displayName: "commentText"
 * }
 * }
 * }).execute(callback)
 * to create a new comment in the feed of the gadget's Jam group
with a callback from a passed
 * in object (activity).
 */
function publishComment() {
    var newComment = $('#comment-text').val();
    if (newComment !== "") {
        var currentComments = getComments();
        currentComments.push(newComment);
        var currentDrafts = getDrafts();
        var updatedDrafts = [];
        $.each(currentDrafts, function(i, val) {
            if (val !== newComment) {
                updatedDrafts.push(val);
            }
        });

        // Updates the map with an array of drafts for the current user
in the private state object (wave)
        wave.getPrivateState().submitDelta({'drafts': updatedDrafts});

        // Updates the map with an array of all user comments from the
public shared object (wave)
        wave.getState().submitDelta({'comments': currentComments});
        // Creates a new feed item from a passed in object (activity).
        // - #{feed_add} gets the message from the English language
locale in spec.xml (line 12).
        // - 'newComment' contains the new comment text
        osapi.activitystreams.create({
            activity: {
                title: "#{feed_add}",
                object: {
                    displayName: newComment
                }
            }
        }).execute(function(result) {});
    }
}

/**
 * Saves a draft for the user.
 * - Uses wave.getPrivateState().submitDelta(map) to update the private
state object (wave) with
 * a passed in map of key-values for the current user. This user-
specific information is

```

```

    *   private and can only be accessed by that user.
    */
    function saveDraft() {
        var currentDrafts = getDrafts();
        currentDrafts.push($('#comment-text').val());

        // Updates the map with an array of drafts for the current user in
the private state object (wave)
        wave.getPrivateState().submitDelta({'drafts': currentDrafts});
    }

    /**
    * Renders updates to 'comments' in the shared state object (wave).
    * - Uses wave.getState().get(map) to get a map of key-values from the
shared state object (wave). All
    *   information is public and can be accessed by all users.
    */
    function publicStateUpdated() {
        // Gets an array of all user comments from the public shared object
(wave)
        if(wave.getState().get('comments')) {
            var comments = getComments();
            if (comments) {
                renderComments(comments);
            }
        }
    }

    /**
    * Renders updates to 'drafts' in the private state object (wave).
    * - Uses wave.getPrivateState().get(map) to get a map of key-values
from the private state object (wave) for the
    *   current user. This user-specific information is private and can
only be accessed by that user.
    */
    function privateStateUpdated() {
        // Gets an array of all drafts for the current user from the private
state object (wave)
        if(wave.getPrivateState().get('drafts')) {
            var drafts = getDrafts();
            if (drafts) {
                renderDrafts(drafts);
            }
        }
    }

    /**
    * Initializes the gadget by:
    * 1. Setting up a callback that executes when the shared state object
(wave) changes.
    *   - Uses wave.setStateCallback(publicCallbackFunction) to setup the
callback for the shared state
    *     object (wave) with a passed-in callback.
    * 2. Setting up a callback that executes when the private state object
(wave) changes.
    *   - Uses wave.setPrivateStateCallback(privateCallbackFunction) to
setup the callback for the private state
    *     object (wave) with a passed-in callback.
    */
    function initGadget() {
        // Checks if the state object (wave) exists and if the gadget is
running in a wave container.
        if (wave && wave.isInWaveContainer()) {

            // Sets up a callback that executes when the shared state object
(wave) changes.
            wave.setStateCallback(publicStateUpdated);

```

```

        // Sets up a callback that executes when the private state object
        (wave) changes.
        wave.setPrivateStateCallback(privateStateUpdated);
    }

    $('#publish-comment').click(function() {
        publishComment();
        $('#comment-text').val("");
    });
    $('#cancel-comment').click(function() {
        $('#comment-text').val("");
    });
    $('#save-draft').click(function() {
        saveDraft();
        $('#comment-text').val("");
    });
    $('#clear-drafts').click(function() {
        clearDrafts();
    });
}

/**
 * Initializes gadget after receiving a notification that the page is
 * loaded and the DOM is ready.
 */
gadgets.util.registerOnLoadHandler(initGadget);
}
};
}();

```

Now that we understand how the gadget works, let's run it.

### RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **spec.xml** with SAP Jam as a **Content** gadget, create a group, add the gadget to that group, and run it. To see the shared communication stream in action, login as another user with another web browser and go to the same group. Try creating and saving comments as drafts as each user to see how the data is stored privately, then publish some of these drafts as comments to see them shared publicly between all users in the comment stream.

Congratulations! This gadget should be up and running!

### Troubleshooting

If you are having any problems registering this gadget with SAP Jam, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

## 4.1.5.3 Advanced Multi-User Sync

In this tutorial we will create a SAP Jam Collaboration OpenSocial gadget to display the context of a statusbar gadget instance.

You will need to perform the following steps before beginning this tutorial:

1. Download the source code from <https://github.com/SAP/SAPJamSampleCode> 📄
2. Extract the zip file and navigate to the gadget folder ( ../OpenSocial/Gadget/context/commentstreamgadgetadvanced )
3. Download the minified jQuery 1.x library from <http://jquery.com/download/> 📄
4. Copy the minified jQuery 1.x library into the gadget's jQuery folder ( ../OpenSocial/Gadget/context/commentstreamgadgetadvanced/vendor/jquery )
5. Download the bootstrap library from <http://getbootstrap.com/getting-started/#download> 📄
6. Extract the zip file and navigate to the css folder ( ../bootstrap-3.x-dist/css )
7. Copy the minified Bootstrap 3.x css file ( bootstrap.min.css ) into the gadget's Bootstrap folder ( ../OpenSocial/Gadget/context/commentstreamgadgetadvanced/vendor/bootstrap )

Once you have completed these steps, open **SAPJAM\_context.html**, adjust the jQuery src filename to match the filename of the minified jQuery 1.x library in your gadget folder and save.

```
<html>
  <head>
    <!--
      * STEP 1
      * - Adjust the jQuery src filename to match the jQuery 1.x version in your
      gadget folder
    -->
    <script type="text/javascript" src="jquery-2.2.3.min.js"></script>
    <script type="text/javascript" src="SAPJAM_context.js"></script>
  </head>
  <body>
</body>
</html>
```

The gadget is now ready to use! Before we test the gadget, let's explore more about how the context of the Content Instance is found using **gadgets.sapjam.context.get**.

Open **SAPJAM\_context.js** and read all of the in-line comments in the code to learn how **gadgets.sapjam.context.get** works within this gadget.

```
function make_SAPJAM_context_Call(){
  gadgets.sapjam.context.get(function(data) {
    console.log(JSON.stringify(data, null, 4));
    /* Begin HTML page */
    var osapiOutput = "";
    osapiOutput += "gadgets.sapjam.context.get --> ";
    /* Adds all the properties of "gadgets.sapjam.context.get" with HTML
    formatting to a string (osapiOutput). */
    osapiOutput += "context = " + data.context;
    osapiOutput += ", id = " + data.id;
    osapiOutput += ", name = " + data.name;
    osapiOutput += ", readOnly = " + data.readOnly;
    /* End HTML page */
    osapiOutput += ". Open your browser console to view the raw JSON
    object.";
    /* Appends the string (osapiOutput) to the body of the HTML page. */
    $("body").append(osapiOutput);
    /* Demonstrates every gadgets.sapjam.statusbar interaction */
    setTimeout(showStatusBar, 1000);
    setTimeout(clearBadgeText, 4000);
    setTimeout(hideStatusBar, 7000);
    setTimeout(clearBadgeText, 10000);
    setTimeout(clickToExpand, 13000);
    setTimeout(clearHighlight, 15000);
  });
}
```

```

}
function showStatusBar(){
    /* Highlights the gadget in blue and makes it blink 3 times */
    gadgets.sapjam.statusbar.highlight();
    /* Expands the gadget and shows the contents of the body element within it */
    gadgets.sapjam.statusbar.show();
    /* Adds a red badge with text to the gadget */
    gadgets.sapjam.statusbar.setBadgeText("Statusbar Expanded");
}
function clearBadgeText(){
    gadgets.sapjam.statusbar.clearBadgeText();
    gadgets.sapjam.statusbar.clearHighlight();
}
function hideStatusBar(){
    gadgets.sapjam.statusbar.hide();
    gadgets.sapjam.statusbar.highlight();
    gadgets.sapjam.statusbar.setBadgeText("Statusbar collapsed");
}
function clickToExpand(){
    gadgets.sapjam.statusbar.highlight();
    gadgets.sapjam.statusbar.setBadgeText("Click here");
}
function clearHighlight(){
    gadgets.sapjam.statusbar.clearHighlight();
}
// Initializes gadget
function init() {
    make_SAPJAM_context_Call();
}
// Initializes gadget after receiving a notification that the page is loaded and
the DOM is ready.
gadgets.util.registerOnLoadHandler(init);

```

Now that we understand how the gadget works, let's run it.

## RUN THIS GADGET!

Upload the source files of this gadget to your publicly accessible server, register the URL of **SAPJAM\_context.xml** with SAP Jam as a **Status Bar** gadget and click **Home**. The gadget will appear in the statusbar at the bottom of the browser window.

**Note:** **Status Bar** gadgets never activate on **Gadget Configuration** pages.

Congratulations! This gadget should be up and running!

## Troubleshooting

If you are having any problems registering this gadget with SAP Jam, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## Reference

To find out more about OpenSocial gadgets, see:

- [OpenSocial Gadget XML Structure \[page 85\]](#)
- [SAP Jam OpenSocial JavaScript API reference \[page 138\]](#)
- <https://developers.google.com/gadgets/docs/ui> .

## 4.1.6 Using OpenSocial Gadgets to extend SAP Jam Collaboration

OpenSocial gadgets enable you to extend SAP Jam Collaboration using the latest technologies, design approaches, and security standards.

### 4.1.6.1 Add Gadget Data to Jam Search

If you know how your gadget stores its key-value data, you can specify slices for SAP Jam Collaboration to add to the search index in addition to the basic gadget instance metadata such as title. When you register a gadget with SAP Jam, you can specify one or more search paths delimited by commas or spaces. SAP Jam extends the local storage mechanisms for appdata, prefs and wave to allow JSON objects to be stored in addition to strings.

When JSON objects are stored in this way, it creates a hierarchy. Use dotted path notation to specify the slice of data you wish to add to the search index. You can use a trailing asterisk in any search path segment as shown in the example below:

Let's create a new gadget, create search paths for it, create a complex JSON object, store it in the gadget's appdata, and then search for data within it.

1. Create a new OpenSocial gadget and name it "Search\_Jam\_Appdata"
2. Create the search paths in the gadget. Enter the following in the "Search Paths" field.

```
*.b.c*
```

3. Save the gadget
4. Go to the "Home" screen.
5. Search SAP Jam for "11Mar1114". No result will be found.
6. Select "Groups" > "Create a Group"
7. Name the group and click the "Create" button. Your group will appear in a few moments.
8. Select "Content"
9. Select "Create"> "Extensions" > "Search\_Jam\_Appdata"
10. Create a title and click the "Create" button.
11. Copy and paste the following JSON object into the gadget's text box:

```
{
  "d": "8Mar524",
  "abc1": "shared_stuff2x",
  "abc2": {
    "b": {
      "c": "11Mar1114"
    }
  }
}
```

12. Click "update appdata". Your JSON object will save to the gadget's appdata local storage.
13. Search SAP Jam for "11Mar1114". No result will be found.
14. Wait 5 minutes for the JSON object to be indexed in SAP Jam.

15. Search SAP Jam for "11Mar1114". If your data is not found, the indexing has not been completed. Wait a minute and try again.
16. Select your group in the search results. Your group and the gadget within will appear.
17. Click "get appdata". Your JSON object will show in the gadget's textbox.

You can download the gadget source code from our official SAP Github repository at:

```
https://github.com/SAP/SAPJamSampleCode .
```

## Run the gadget

Upload the source files of our gadget to your publicly accessible server, register the URL of Search\_Jam\_Appdata.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

```
https://sapjamsamplecode.github.io/OpenSocial/Gadget/Search/Search\_Jam\_Appdata/Search\_Jam\_Appdata.xml .
```

Congratulations! Our gadget should be up and running!

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see: [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### 4.1.6.2 Applying String Localization to OpenSocial Gadgets

Localization is supported via JSPEL (Java Server Pages Expression Language) which supports embedded and remote locales. Define or reference message resources, then use JSPEL expressions rooted at Msg, and then extend by the msg name where you need the localized value.

Applying localization to a gadget's xml file involves the following steps:

1. Define your locales.

```
<Locale lang="en"></Locale>
<Locale lang="fr"></Locale>
```

2. Define the key for each locale message.

```
<Locale lang="en">
  <msg name="detectedLocaleGreeting"></msg>
</Locale>
<Locale lang="fr">
  <msg name="detectedLocaleGreeting"></msg>
```



```
</Locale>
```

3. Define the messages for each locale.

```
<Locale lang="en">
  <msg name="detectedLocaleGreeting">Detected Locale = <b>English</b><br/>
  Greeting = <b>Hello!</b></msg>
</Locale>
<Locale lang="fr">
  <msg name="detectedLocaleGreeting">Detected Locale = <b>French</b><br/>
  Greeting = <b>Bonjour!</b></msg>
</Locale>
```

4. Embed your locale object (Msg.KEY) into HTML.

```
${Msg.detectedLocaleGreeting}
```

5. Your locales, key, messages for each locale, locale object and the HTML embedding of your locale object will now be setup as shown below:

Locales:	en=English, fr=French
KEY:	detectedLocaleGreeting
XML:	<msg name="KEY">DATA</msg>
Object:	Msg.KEY=DATA
Relationship (XML = Object):	'<msg name="KEY">DATA</msg>' =
'Msg.KEY=DATA'	
Object embedded in HTML:	\${Msg.KEY}

A sample OpenSocial String Localization gadget that uses the locale information from the previous steps is shown below:

```
<Locale lang="en">
<?xml version="1.0" encoding="UTF-8" ?>
<Module specificationVersion='2'>
  <ModulePrefs title="String Localization"
    height="250"
    author="SAP"
    description="This gadget localizes its output based on a
compatible
detected locale (English, French).">
    <Link rel="mediumIcon" href="mediumIcon.png" />
    <Link rel="icon" href="icon.png" />
    <Require feature="osapi" />
    <Require feature="dynamic-height"/>
    <Require feature="minimessage"/>
  <!--
  Locales:
  KEY:
  XML:
  Object:
  Relationship (XML = Object):
'Msg.KEY=DATA'
  Object embedded in HTML:
  -->

  <Locale lang="en">
    <msg name="detectedLocaleGreeting">Detected Locale = <b>English</b><br/>
    Greeting = <b>Hello!</b></msg>
  </Locale>
  <Locale lang="fr">
    <msg name="detectedLocaleGreeting">Detected Locale = <b>French</b><br/>
    Greeting = <b>Bonjour!</b></msg>
  </Locale>
</ModulePrefs>
<Content type="html">
```

```

<![CDATA[
  <h1>String Localization</h1>
  <p>This gadget localizes its output based on a detected locale.</p>
  <p>Adjust your browser to a different locale (English or French) and
    reload this gadget.
    The 'Detected Locale' and 'Greeting' will automatically change.</p>
  <p>${Msg.detectedLocaleGreeting}</p>
]]>
</Content>
</Module>

```

You can download the gadget source code from our official SAP Github repository at:

<https://github.com/SAP/SAPJamSampleCode> ➡

## Run the gadget

Upload the source files of our gadget to your publicly accessible server, register the URL of Localization.xml with SAP Jam Collaboration, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

<https://sapjamsamplecode.github.io/OpenSocial/Gadget/Localization/Localization.xml> ➡

The gadget should be up and running.

## Troubleshooting

If you are having any problems registering our gadget with your SAP Jam Collaboration live service, see: [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

### 4.1.6.3 Applying Responsive Design to OpenSocial Gadgets

Gadgets in a SAP Jam Collaboration page will change their width depending on how the container is sized, which is dependant on window width. Currently, gadget containers vary dynamically in size from approximately 650 to 850 pixels. These sizes are subject to change. You can use a responsive library such as Twitter Bootstrap or Zurb Framework that uses relative sizing and media queries for breakpoints. Alternatively consider using percentage widths directly.

You will not be able to set viewport meta directives in the head element since you won't have direct access to it in the gadget's XML metadata file.

## 4.1.6.4 Register OpenSocial Gadgets with OAuth

OpenSocial gadgets can be registered with OAuth configurations so that makeRequest proxy calls can be authenticated. You can have multiple configurations for both OAuth 1.0a and OAuth 2.0. For each configuration, specify the service name, consumer key/client id, consumer secret/client secret and signature method as required. In particular, the service name should match the parameter `gadgets.io.RequestParameters.OAUTH_SERVICE_NAME` used in the gadget's call to `gadgets.io.makeRequest`.

### Using OAuth with Internet Explorer

Internet Explorer requires the OAuth URIs to be added to the **Trusted Sites** list as shown below:

- `https://<your_SAP_Jam_URI>`
- `https://<external_OAuth_URI>`

For example, to use OAuth with SAP Jam Collaboration and Twitter add the following URIs to the **Trusted Sites** list:

- `https://*.sapjam.com`
- `https://*.twitter.com`

## 4.1.6.5 OAuth 2.0 SAML Bearer Assertion Flow for OpenSocial Gadgets

This flow provides OpenSocial Gadgets access to external web services via SSO authentication using OAuth 2 and SAML Bearer Assertion.

The three components involved are:

- The OpenSocial Gadget in SAP Jam Collaboration. This is the custom HTML5/Javascript program the user interacts with.
- The OpenSocial OAuth 2.0 Service Configuration. This enables SSO authentication between the OpenSocial Gadget and the 3rd party OData service using:
  1. OAuth 2 authentication (key and secret from the OAuth client on the 3rd party OData service)
  2. SAML Bearer Assertion (identity from a trusted identity provider)
- The 3rd party OData service (for example, SAP Jam OData API). This provides a response with the resources requested (JSON, XML, etc.) by the OpenSocial Gadget.

These components interact with each other in the following order:

1. The OpenSocial Gadget makes a `gadgets.io.makeRequest` call through SAP Jam's OpenSocial OAuth 2.0 Service Configuration to the 3rd party OData service.
2. The OpenSocial OAuth 2.0 Service Configuration intercepts the `gadgets.io.makeRequest` call and provides SSO authentication between the OpenSocial Gadget and the 3rd party OData service using:
  - OAuth 2 authentication (key and secret from the OAuth client on the 3rd party OData service)

- SAML Bearer Assertion (identity from a trusted identity provider)
- 3. The 3rd party OData service (for example, SAP Jam OData API) provides a response with the resources requested (JSON, XML, etc.) by the OpenSocial Gadget.

## Example - OpenSocial Gadget Calling the SAP Jam API

To demonstrate this flow we will register an OpenSocial Gadget to show all groups you have access to in your SAP Jam instance using the SAP Jam OData API. Your instance of SAP Jam will:

- Register the OpenSocial Gadget.
- Provide SSO Authentication through the OpenSocial OAuth 2.0 Service Configuration using:
  - a SAP Jam OAuth client (provides key and secret)
  - a SAP Jam Local Identity Provider (provides identity via SAML Bearer Assertion)
  - a SAP Jam SAML Trusted Identity Provider (sets up a trust relationship between SAP Jam and the SAML Local Identity Provider)
- Provide the OData service.

### 4.1.6.5.1 Example - Gadget Calling SAP Jam Collaboration API Setup

To setup this workflow you must perform the following steps:

#### 4.1.6.5.1.1 1. Download SAML Sample Code

1. Download the gadget source code from our official SAP Github repository at: <https://github.com/SAP/SAPJamSampleCode> .
2. Extract the zip file.
3. Download the minified Mithril library (mithril.min.js) from <http://mithril.js.org/installation.html> .
4. Copy the minified Mithril library (mithril.min.js) into the gadget's root folder (../OpenSocial/Gadget/SAMLBearerAssertion).
5. Upload the gadget's root folder (../OpenSocial/Gadget/SAMLBearerAssertion) to your webhost.

#### 4.1.6.5.1.2 2. Configure your SAP Jam Collaboration SAML Trusted Identity Provider

1. From the **Admin** console sidebar menu, select **Integrations** > **SAML Local Identity Provider** .
2. Copy all content from the **Issuer** field.

3. Open **SAML Trusted IDPs** in a new tab.
4. Click **Register your SAML Trusted IDP**.
5. Paste it into the **IPD ID** field.
6. Switch to the **SAML Local Identity Provider** tab.
7. If a X509 certificate does not exist, click **Generate Key Pair** and click **Save changes**. Copy the certificate from the **X509 Certificate (Base64)** field.
8. Switch to the **Register a new SAML Trusted Identity Provider** tab.
9. Paste the certificate into the **X509 Certificate (Base64)** field.
10. Select **Company** from the **Administrative Area** drop-down list.
11. Click **Register**.

### 4.1.6.5.1.3 3. Configure your SAP Jam Collaboration OAuth Client

1. From the **Admin** console sidebar menu, select **Integrations > OAuth Clients**.
2. Click **Add OAuth Client**.
3. Create a name for your OAuth Client in the **Name** field.
4. Enter your Jam instance base URL in the **Integration URL** field:

```
https://{jam_instance}.sapjam.com
```

5. Click **Save**. Your OAuth client appears in the OAuth Clients list.
6. Click **View** on your OAuth Client.
7. The **client ID (Key)** and **Secret** of your OAuth Client appears.

### 4.1.6.5.1.4 4. Configure your SAP Jam Collaboration OpenSocial Gadget Configuration

1. [Register your gadget with SAP Jam Collaboration \[page 74\]](#).
2. Add an OAuth 2.0 Configuration for your gadget:
  1. Click **Add Service Configuration** in **OAuth 2.0 Service Configurations**.
  2. In the **Service Name** text field, paste the **name** of your SAP Jam **OAuth Client**.
  3. In the **Client Id** text field, paste the **key** of your SAP Jam **OAuth Client**.
  4. In the **Client Secret** text field, paste the **secret** of your SAP Jam **OAuth Client**.
  5. From **Grant Type**, select **SAML 2.0 Bearer Assertion** (`urn:ietf:params:oauth:grant-type:saml2-bearer`).
  6. From **SAML 2.0 Assertion Audience**, select **Token endpoint URL of the authorization server**.
3. Click **Save**.

## 4.1.6.5.1.5 5. Configure your OpenSocial Gadget

1. Open "`\OpenSocial\Gadget\SAMLEntity\SAMLEntity.xml`". Make sure that the:
  - **Service name** in this file matches the **Service Name** in your gadget's **Service Configuration**.
  - **Token url** value in this file contains this URL:
    - "`https://<YOUR_SAP_JAM_INSTANCE>.sapjam.com/api/v1/auth/token`".
2. Open "`\OpenSocial\Gadget\SAMLEntity\SAMLEntity.js`". Make sure that the:
  - **OAUTH\_SERVICE\_NAME** in this file matches the **Service Name** in your gadget's **Service Configuration**.
  - OData call contains the following URL within the **makeRequest** parameter:
    - "`https://<YOUR_SAP_JAM_INSTANCE>.sapjam.com/api/v1/OData/Groups?$format=json`".
3. Upload these files to your webhost.

## 4.1.6.5.1.6 6. Test your Gadget

1. Switch to your gadget configuration in SAP Jam Collaboration.
2. Click **Reload Gadget Source**. The gadget is displayed with a list of the groups you can access in your SAP Jam instance.

## 4.1.7 SAP Jam OpenSocial JavaScript API reference

The majority of the work that must be done to create an SAP Jam Collaboration OpenSocial gadget is to develop the JavaScript business logic. Several JavaScript API libraries exist for developing OpenSocial Gadgets. This section begins with a short tutorial on developing the JavaScript for your gadget, and it provides documentation of the API calls or endpoints that are supported by the SAP Jam Collaboration implementation of the OpenSocial API.

### 4.1.7.1 OpenSocial features supported in the SAP Jam Collaboration implementation

This page provides an overview of the SAP Jam Collaboration support for various OpenSocial API calls.

#### Supported OpenSocial Core Gadget API features

The following table shows which OpenSocial Core Gadget API features are supported in the SAP Jam implementation.

## OpenSocial Core Gadget Supported Features

Feature	Support	Description	Available Functions
io	yes	Retrieves remote content.	encodeValues, getProxyUrl, makeRequest, and an explanation of the request parameters used.
json	yes	Translates objects to and from JSON.	parse and stringify
util	yes	Utility functions for escaping and unescaping strings, sanitizing HTML, and registering on-load handlers.	escapeString, getFeatureParameters, hasFeature, registerOnLoadHandler, sanitizeHtml, and unescapeString.
views	partial	Allows you to run different code depending on the "view" value, which can be set to "home" (for Content gadgets) or "profile" (for Profile gadgets).	TBD.
MiniMessage	yes	Creates and displays messages within a gadget, including status and error messages.	Constructor, createDismissibleMessage, createTimerMessage, createStaticMessage, and dismissMessage.
TabSet	no	Adds tabs to a gadget and manages them.	Constructor, addTab, alignTabs, displayTabs, getHeaderContainer, getSelectedTab, getTabs, removeTab, setSelectedTab, and swapTabs.
Tab	yes	Manages specific tabs in a gadget.	getCallback, getContentContainer, getIndex, getName, and getNameContainer.
flash	yes	Embeds Flash content within a gadget.	embedFlash, embedCachedFlash, and getMajorVersion.
window	yes	Gets information about, and allows you to modify, the window the gadget is placed in.	adjustHeight, getViewportDimensions, and setTitle.
skins	yes	Developer information required.	TBD.
rpc	yes	Developer information required.	TBD.
oauth	partial	Developer information required.	TBD.
pubsub	no	Not currently supported.	N.A.
actions	no	Not currently supported.	N.A.
selection	no	Not currently supported.	N.A.
Hub	no	Not currently supported.	N.A.

For more information, see:

- [Gadgets API](#)  

- [OpenSocial Core Gadget Specification 2.5](#) 

## Supported OpenSocial Social Gadget API features

The following table shows which OpenSocial Social Gadget API features are supported in the SAP Jam implementation.

**OpenSocial Social Gadget Supported Features**

Feature	Support	Description	Available Functions
people	yes	Gets information about users.	get (user), getViewer, getViewerFriends, getOwner, and getOwnerFriends.
appdata	yes	Retrieves, updates, and deletes a user's data.	get, update, and delete.
activities	no	Not currently supported.	N.A.
groups	no	Not currently supported.	N.A.
messages	no	Not currently supported.	N.A.
albums	no	Not currently supported.	N.A.

For more information, see:

- [OpenSocial Social Gadget Specification 2.5](#) 



## Supported Wave API features

The following table shows which Wave API features are supported in the SAP Jam implementation.

**Wave API Supported Features**

Feature	Support	Description	Available Functions
Participants	yes	Provides "soft real-time" updates for OpenSocial gadget content.	setStateCallback(callback, opt_context), setPrivateStateCallback(callback, opt_context), setParticipantCallback(callback, opt_context), getState(), getPrivateState(), and submitDelta(delta).

For more information, see:

- [Google Wave Gadgets API](#) 
- [Apache Incubator Wave](#) 



## 4.1.7.2 SAP Jam Collaboration OpenSocial Coding Options Comparisons

As there are several JavaScript libraries supported that have some overlapping functionality, you have certain choices that you must make for some operations. This page reviews some significant characteristics of those options to help you to make effective choices.

### osapi.Appdata versus wave versus userPrefs

The major advantage of `wave` over `appdata` and `userPrefs` is real time support. However, another difference to consider before deciding on the implementation is data permissions:

- `wave` — supports private and public states which means *shared-read-write* or *private-read-write*.
- `UserPrefs` (XML) — supports only *private-read-write*, which is mostly used for user personal settings of the gadget.
- `appdata` — supports *shared-read-private-write* which can be a desired functionality, but which is not supported by either `wave` or `UserPrefs`.

### osapi.People versus wave.Participants

Both `osapi.People` and `wave.Participants` have the overlapping capabilities of getting member IDs, thumbnails, and names.

- If you want to have real time updates of changes to the "participants", you should choose `wave.Participants` as it provides real time updates using `ParticipantCallback`.
- If you want to have more information about SAP Jam Collaboration users than their ID, thumbnail, and name, you should choose `osapi.people`.

Note that these APIs can complement each other. For example, you can use `wave.Participants` for real time support and to extract participant IDs, and then you can pass the participant IDs on to `osapi.People` to get more information on the participants.

## 4.1.7.3 OpenSocial Gadgets API reference

This API represents the core functionality of a (potentially non-"social" gadget). This is sometimes referred to by its namespace, the `gadgets.*` API.

## 4.1.7.3.1 gadgets.flash

Allows you to embed Flash content within a gadget.

**Required feature:** flash

### gadgets.flash.embedFlash

Embeds a Flash file within a gadget.

#### Parameters

Name	Description
swfUrl	The URL of the SWF file to be embedded.
swfContainer	Either the ID or object reference of the HTML container element in which to embed to content.
swfVersion	The minimum Flash Player version required.
opt_params	( <i>optional</i> ) An object that may contain any valid HTML parameters.

#### Returns

Type	Description
Boolean	Whether the function call completed successfully.

#### Example

```
function show() {
    var url = "https://mydomain.com/flash/sample.swf";
    gadgets.flash.embedFlash(url, "flashcontainer", {
        swf_version: 6,
        id: "flashid",
        width: 400,
        height: 250
    })
}
```

### gadgets.flash.embedCachedFlash

Embeds a cached Flash file within a gadget.

#### Parameters

Name	Description
swfUrl	The URL of the SWF file to be embedded.
swfContainer	Either the ID or object reference of the HTML container element in which to embed to content.
swfVersion	The minimum Flash Player version required.

opt_params	( <i>optional</i> ) An object that may contain any valid HTML parameters.
Returns	
Type	Description
Boolean	Whether the function call completes successfully.
Example	
<pre>function show() {     var url = "https://mydomain.com/flash/sample.swf";     gadgets.flash.embedCachedFlash(url, "flashcontainer", {         swf_version: 6,         id: "flashid",         width: 400,         height: 250     }) }</pre>	

## gadgets.flash.getMajorVersion

Detects the Flash player and its major version.

Parameters	
Name	Description
None.	N.A.
Returns	
Type	Description
Number	The major version of the Flash player.
Example	
<pre>function displayInfo() {     document.getElementById("flashcontainer").innerHTML = "Flash Version: " +     gadgets.flash.getMajorVersion() + "."; }</pre>	

### 4.1.7.3.2 gadgets.io

Provides operations for retrieving remote content.

**Required feature:** None

## gadgets.io.AuthorizationType

Used by gadgets.io.RequestParameters.AUTHORIZATION.

### Parameters

Name	Description
NONE	No authorization.
OAUTH	The container will use OAuth 1.0A for authentication.
OAUTH2	The container will use OAuth 2.0 for authentication.
SIGNED	The request will be signed by the container.

### Example

```
var opt_params = {};  
opt_params[gadgets.io.RequestParameters.AUTHORIZATION] =  
gadgets.io.AuthorizationType.OAUTH;
```

## gadgets.io.ContentType

Used by gadgets.io.RequestParameters.CONTENT\_TYPE.

### Parameters

Name	Description
DOM	Returns a DOM object; used for fetching XML.
FEED	Returns a JSON representation of an RSS or Atom feed.
JSON	Returns a JSON object.
TEXT	Returns text; used for fetching HTML.

### Example

```
var opt_params = {};  
opt_params[gadgets.io.RequestParameters.CONTENT_TYPE] =  
gadgets.io.ContentType.JSON;
```

## gadgets.io.encodeValues

Converts an input object into a URL-encoded data string.

### Parameters

Name	Description
------	-------------

fields	An object containing the form fields that you want to encode.
Returns	
Type	Description
String	The processed post data, including a trailing ampersand (&).
Usage	<static> Type: {String} gadgets.io.encodeValues(fields)
Example	
<pre>function makePOSTRequest(url, postdata) {     var opt_params = {};     postdata = gadgets.io.encodeValues(postdata);     opt_params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.POST;     opt_params[gadgets.io.RequestParameters.POST_DATA]= postdata;     gadgets.io.makeRequest(url, responsePOST, opt_params); };</pre>	

## gadgets.io.getProxyUrl

Gets the proxy version of the passed-in URL. gadgets.io.getProxyUrl can be used to cache the resources used in a gadget and allow the gadget to render faster.

Parameters	
Name	Description
url	The URL to get the proxy URL for, as a string.
Returns	
Type	Description
String	The proxy version of the URL.
Usage	<static> Type: {String} gadgets.io.getProxyUrl(url, opt_params)
Example	
<pre>var img = document.createElement("img"); img.src = gadgets.io.getProxyUrl("http://mydomain.com/images/sample_image.png");</pre>	

## gadgets.io.makeRequest

Retrieves content from the provided URL and feeds the content into a callback function.

Parameters	
Name	Description

url	<b>Type:</b> {String}  The URL where the content is located.
callback	<b>Type:</b> {Function}  The function to call with the data retrieved from the URL. The callback function isn't called until after the existing callstack has completed execution.
opt_params	<b>Type:</b> {Map.<gadgets.io.RequestParameters Object>}  Additional request parameters or proxy request parameters.

Returns

Type	Description
Object	<p>A JavaScript object that contains the following properties:</p> <ul style="list-style-type: none"> <li>• <b>data:</b> Parsed data of the response, if applicable. This will contain a different type of data depending on the type of request that was made. See the <code>ContentType</code> parameters for information about what to expect in this field. The raw response text is returned if the response could not be parsed.</li> <li>• <b>errors:</b> This will contain an array of any errors that occurred when making this request. For example, if a 500 error occurred in the request: [ "500 error" ]</li> <li>• <b>headers:</b> An object containing the response headers. Header names are used as the keys for this object.</li> <li>• <b>rc:</b> This is a numeric value representing the HTTP status code of the response.</li> <li>• <b>text:</b> This will return the unparsed text of the response.</li> <li>• <b>oauthApprovalUrl:</b> If this value is specified, the user needs to visit an external page to approve the gadget's request to access data. Use of a pop-up window to direct the user to the external page is recommended. Once the user has approved access, the gadget can repeat the <code>makeRequest</code> call to retrieve the data.</li> <li>• <b>oauthError:</b> If this value is specified, it indicates an OAuth-related error occurred. The value will be one of a set of string constants that can be used for programmatically detecting errors.</li> <li>• <b>oauthErrorText:</b> If this value is specified, it indicates an OAuth-related error occurred. The value is free-form text that can be used to provide debugging information for gadget developers.</li> </ul>
Usage	<code>&lt;static&gt; gadgets.io.makeRequest(url, callback, opt_params)</code>

Example 1 - Making a DOM request.

```
function makeDOMRequest() {
    var opt_params = {};
    opt_params[gadgets.io.RequestParameters.CONTENT_TYPE] =
gadgets.io.ContentType.DOM;
    var url = "http://mydomain.com/data/data.xml";
    gadgets.io.makeRequest(url, callback, opt_params);
};
function response(obj) {
    //do something here
}
makeDOMRequest();
```

Example 2 - Making an OData call to the SAP Jam Collaboration API using SAMLBearerAssertion. [\[page 135\]](#)

```

/*
Gadget Source: https://github.com/SAP/SAPJamSampleCode/tree/master/OpenSocial/
Gadget/SAMLEBearerAssertion
gadgets.io.makeRequest(url, callback, opt_params);
url: OData Call URL
callback: Callback function used to process the response.
opt_params: Additional OData proxy request parameters (shown below):
    AUTHORIZATION: The type of authentication to use when fetching the content.
    OAUTH_SERVICE_NAME: The nickname the gadget uses to refer to the OAuth
    <Service> element from its XML spec.
    CONTENT_TYPE: The type of content to retrieve at the specified URL.
*/
{
    var self = this;
    gadgets.io.makeRequest("https://CALL_TO_YOUR_ODATA_SERVICE_PROVIDER",
        function(result) {
            console.log(result);
            self.setState({data: result.data, users: self.state.users});
        },
        {
            AUTHORIZATION: 'OAUTH2',
            OAUTH_SERVICE_NAME: 'YOUR_OAUTH_SERVICE_NAME',
            CONTENT_TYPE: gadgets.io.ContentType.JSON
        }
    );
}

```

## gadgets.io.MethodType

Used by gadgets.io.RequestParameters.METHOD

### Parameters

Name	Description
DELETE	Container support for this method type is OPTIONAL.
GET	The default type.
HEAD	Container support for this method type is OPTIONAL.
POST	Container support for this method type is OPTIONAL.
PUT	Container support for this method type is OPTIONAL.

### Example

```

var opt_params = {};
opt_params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.GET;

```

## gadgets.io.ProxyUrlRequestParameters

Used by gadgets.io.getProxyUrl() method.

#### Parameters

Name	Description
REFRESH_INTERVAL	Explicitly sets the lifespan of cached content. The Refresh Interval is the number of seconds the container should cache the given response. By default, the HTTP caching headers will be respected for fetched content. If the refresh interval is set, this value will take precedence over any HTTP cache headers. If this value is not set and there are no HTTP caching headers specified, this value will default to 3600 (one hour). Note that Signed requests and objects with POST_DATA present will generally not be cached. This field may be used interchangeably with the string 'REFRESH_INTERVAL'.

## gadgets.io.RequestParameters

Used by `gadgets.io.makeRequest`.

#### Parameters

Name	Description
ALIAS	Specifies the 'Alias' request parameter as defined by Core-Data Request Parameters.
AUTHORIZATION	The type of authentication to use when fetching the content. The default is Authorization. Type: NONE. Accepted values are: <ul style="list-style-type: none"> <li>NONE: No authorization.</li> <li>OAuth: Specifies OAuth authentication.</li> </ul>
CONTENT_TYPE	The type of content to retrieve at the specified URL. The default value is ContentType. TEXT. Accepted values are: <ul style="list-style-type: none"> <li>DOM: Retrieves a DOM object. Used for fetching XML.</li> <li>FEED: Retrieves a JSON representation of an RSS or Atom feed.</li> <li>JSON: Retrieves a JSON object.</li> <li>TEXT: Retrieves text. Used for fetching HTML.</li> </ul>
GET_FULL_HEADERS	Returns the full headers from the response.
GET_SUMMARIES	If the content is a feed, specifies whether to fetch summaries for the feed. The default value is false.
HEADERS	The HTTP headers to send to the URL, specified as a Map. <String,String>. Defaults to null.
METHOD	The method to use when fetching the remote content. Accepted values are: <ul style="list-style-type: none"> <li>DELETE</li> <li>GET</li> <li>HEAD</li> <li>POST</li> <li>PUT</li> </ul>
NUM_ENTRIES	If the number is a feed, specifies the number of feed entries to retrieve. The default value is 3.
OAuth_RECEIVED_CALLBACK	The URL to return after authorization.



OAUTH_REQUEST_TOKEN	A service provider may be able to automatically provision a gadget with a request token that is preapproved for access to a resource. The gadget can use that token with the OAUTH_REQUEST_TOKEN parameter. This parameter is optional.
OAUTH_REQUEST_TOKEN_SECRET	The secret corresponding to a preapproved request token. This parameter is optional.
OAUTH_SERVICE_NAME	The nickname the gadget uses to refer to the OAuth <Service> element from its XML spec. If unspecified, defaults to "".
OAUTH2_SCOPE	The value of the OAuth 2.0 scope parameter to be used with this request.
OAUTH_TOKEN_NAME	The nickname the gadget uses to refer to an OAuth token granting access to a particular resources. If unspecified, defaults to "". Gadgets can use multiple token names if they have access to multiple resources from the same service provider. For example, a gadget with access to a contact list and a calendar might use a token name of "contacts" to use the contact list token, and a contact list of "calendar" to use the calendar token.
OAUTH_USE_TOKEN	<p>This parameter can be either "never", "if_available", or "always". Some service provider APIs do not require an OAuth access token. Such APIs authenticate the calling application via the OAuth consumer key, and then allow the request to proceed. You can set OAUTH_USE_TOKEN to "never" to avoid unnecessarily requesting the user's approval to access such an API.</p> <p>Some OAuth APIs provide limited data if a user has not granted approval, but can offer additional functionality if a user has granted the calling application an access token. If you are using an API that can accept an OAuth access token, but does not require the token, you can set OAUTH_USE_TOKEN to "if_available". If the user has already approved access, the access token will be sent. If the user has not approved access, the request will proceed without an access token.</p> <p>Many OAuth APIs only function if an access token is sent. You must ask for the user's approval before using such an API. Set OAUTH_USE_TOKEN to "always" to require that an access token be available. If no access token is available, the approval URL will be returned to your gadget.</p> <p>If you set AUTHORIZATION to SIGNED, the default value for OAUTH_USE_TOKEN is "never". If you set AUTHORIZATION to OAUTH, the default value for OAUTH_USE_TOKEN is "always".</p>
POST_DATA	The data to send to the URL using the POST method. Defaults to null.
REFRESH_INTERVAL	Sets the lifespan, in seconds, of cached content. If the refresh interval is set, this value will take precedence of any HTTP cache headers. If this value is not set, and no HTTP caching headers are specified, this value defaults to 3600 seconds (one hour).

#### Example

```
var opt_params = {};
opt_params[gadgets.io.RequestParameters.CONTENT_TYPE] =
gadgets.io.ContentType.JSON;
opt_params[gadgets.io.RequestParameters.AUTHORIZATION] =
gadgets.io.AuthorizationType.OAUTH;
opt_params[gadgets.io.RequestParameters.OAUTH_SERVICE_NAME] = "twitter";
opt_params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.GET;
gadgets.io.makeRequest(url, callback, opt_params);
```

### 4.1.7.3.3 gadgets.json

Provides operations for translating objects to and from JSON.

**Required feature:** None.

#### gadgets.json.parse

Parses a JSON string and converts it into a JavaScript object.

##### Parameters

Name	Description
text	The string to convert into an object.

##### Returns

Type	Description
Object	The object parsed from the string that is passed to the function call.

##### Example

```
// Convert JSON string into an object
function toObject(str) {
    return gadgets.json.parse(str);
}
```

#### gadgets.json.stringify

Converts a JavaScript object to a JSON string.

##### Parameters

Name	Description
object	The object to convert to a string.

##### Returns

Type	Description
String	The JSON string.

##### Example

```
// Encode object as JSON string
function toJSON(obj) {
    return gadgets.json.stringify(obj);
}
```

## 4.1.7.3.4 gadgets.Minimessage

Provides functions for creating and displaying messages within a gadget, including status and error messages.

**Required feature:** minimessage.

### Constructor

Creates a MiniMessage class that can be used to create messages that are displayed within the gadget.

#### Parameters

Name	Description
opt_moduleId	( <i>optional</i> ) The ModuleId, as a string.
opt_container	( <i>optional</i> ) The HTML element where the messages are to be displayed. If not specified, messages will appear at the top of the gadget.

#### Returns

Type	Description
None.	N.A.

#### Example

```
gadgets.Minimessage(opt_moduleId, opt_container)
```

For a more extensive example of the use of MiniMessage, including the constructor, `createStaticMessage`, and `dismissMessage`, see [Full MiniMessage Code Example \[page 154\]](#).

### createDismissibleMessage

Creates a dismissible message with an [x] icon that allows users to dismiss the message. When the message is dismissed, the optional callback function is called.

#### Parameters

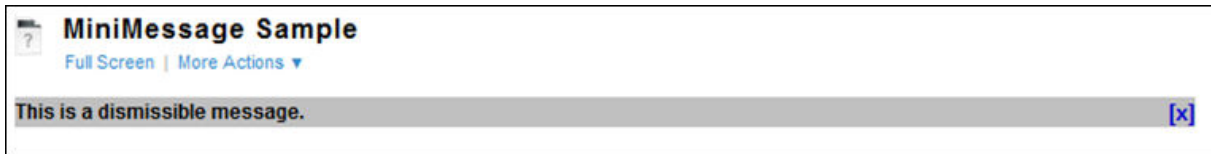
Name	Description
message	The message that is to be displayed, either as a string or as a DOM element.
opt_callback	( <i>optional</i> ) A callback function that is called when the message is dismissed.

#### Returns

Type	Description
HTMLElement	The HTML element of the created message.

#### Example

```
var msg = new
gadgets.MiniMessage( __MODULE_ID__, document.getElementById("messageBox"));
msg.createDismissibleMessage("This is a dismissible message.");
```



## createTimerMessage

Creates a message that is displayed for the specified number of seconds. When the timer expires, the message is dismissed and the optional callback function is executed.

### Parameters

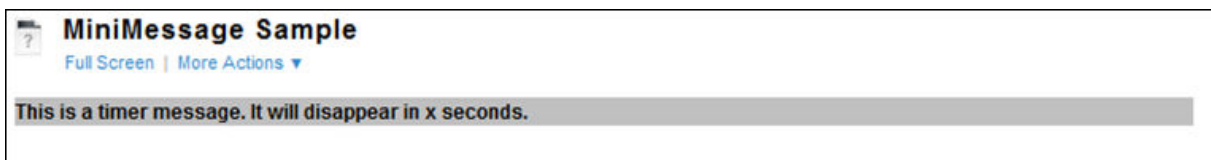
Name	Description
message	The message that is to be displayed, either as a string or as a DOM element.
seconds	The number of seconds to wait before the message is dismissed.
opt_callback	( <i>optional</i> ) A callback function that is called when the message is dismissed.

### Returns

Type	Description
HTMLElement	The HTML element of the created message.

### Example

```
var msg = new
gadgets.MiniMessage( __MODULE_ID__, document.getElementById("messageBox"));
msg.createTimerMessage("This is a timer message.", 5);
```



## createStaticMessage

Creates a static message that can only be dismissed programmatically, by calling the `dismissMessage` function.

#### Parameters

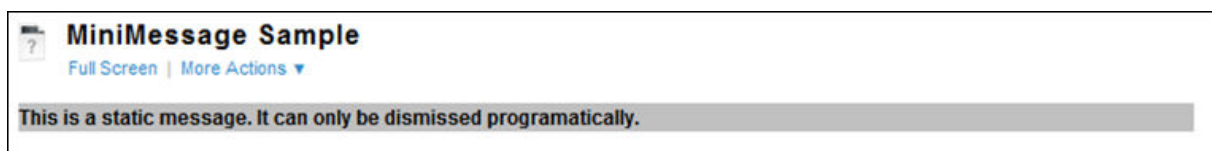
Name	Description
message	The message that is to be displayed, either as a string or as a DOM element.

#### Returns

Type	Description
HTMLElement	The HTML element of the created message.

#### Example

```
var msg = new
gadgets.MiniMessage(__MODULE_ID__, document.getElementById("messageBox"));
msg.createStaticMessage("This is a static message.");
```



For a more extensive example of the use of MiniMessage, including the constructor, `createStaticMessage`, and `dismissMessage`, see [Full MiniMessage Code Example \[page 154\]](#).

## dismissMessage

Dismisses the specified static message.

#### Parameters

Name	Description
HTMLElement	The HTML element of the message to dismiss.

#### Returns

Type	Description
None.	N.A.

#### Example

```
function dismiss_message() {
    msg.dismissMessage(messageContainer);
}
```

For a more extensive example of the use of MiniMessage, including the constructor, `createStaticMessage`, and `dismissMessage`, see [Full MiniMessage Code Example \[page 154\]](#).

## Full MiniMessage Code Example

This section displays a code example that demonstrates an extensive use of the MiniMessage functions.

```
var msg = new gadgets.Minimessage( __MODULE_ID__,
document.getElementById("messageBox"));
var messageContainer;
function create_message() {
    messageContainer = msg.createStaticMessage("This is a static message");
}
function dismiss_message() {
    msg.dismissMessage(messageContainer);
}
```

### 4.1.7.3.5 gadgets.sapjam

Provides operations for interacting with SAP Jam.

**Required feature:** None

#### gadgets.sapjam.alerts.showToast

Displays a toast notification on the current Jam page. Can be configured to show a normal or error state.

Parameters

Name	Description
alertMessage (string)	The text string to display within a toast.
options (object)	An optional parameter that contains the following properties: <ul style="list-style-type: none"><li>error (boolean) - Defines whether the toast should be displayed in an error state or not. While a normal message will dismiss itself within 5 seconds, an error message requires the user to explicitly close the notification.</li></ul>

#### gadgets.sapjam.asyncMessage.get

Retrieves data stored by the gadget container through gadgets.sapjam.asyncMessage.post

Parameters

Name	Description
token	The identifier of the stored message.

callback	Called by the gadget container when the data is retrieved. Contains the following properties: <ul style="list-style-type: none"> <li>object - The requested data stored by the container. It can be null if the requested identifier does not exist.</li> </ul>
----------	---

## gadgets.sapjam.asyncMessage.post

Sends arbitrary data to the gadget container, which would store it and send back a receipt token via a callback.

### Parameters

Name	Description
object	An object containing data to be stored by the container.
callback	Called by the gadget container when the data is received and saved. Contains the following properties: <ul style="list-style-type: none"> <li>token - The identifier of the stored message.</li> </ul>

## gadgets.sapjam.context.get

Returns metadata about the current gadget instance.

### Parameters

Name	Description
callback	Receives context data for the gadget instance. Contains the following properties: <ul style="list-style-type: none"> <li>context - Object that contains the following properties: <ul style="list-style-type: none"> <li>id: The unique identifier of the gadget instance</li> <li>context: The gadget type, could be of ['context', 'profile', 'statusbar', 'unknown']</li> <li>name: The name of the gadget instance</li> <li>readOnly: This is True if the current user has no write access to gadget data, otherwise False.</li> </ul> </li> </ul>

## gadgets.sapjam.data.getInitializingData

Retrieves data that should be used to setup the gadget when it is first created. Currently, only Group Gadgets have initializing data.

### Parameters

Name	Description
callback	Receives the data for the gadget instance. Contains the following properties: <ul style="list-style-type: none"> <li>data - Initializing data that should be used by the gadget when it is first created.</li> </ul>

## gadgets.sapjam.dialog.confirm

Creates a simple dialog with a confirmation and cancel button. An optional callback can be given to handle the user's response to the confirmation dialog.

### Parameters

Name	Description
options	An object with the following optional properties: <ul style="list-style-type: none"><li>• title - Dialog box title.</li><li>• message - Dialog box message.</li><li>• okLabel - Confirmation button label.</li><li>• cancelLabel - Cancellation button label.</li></ul>
callback	A function that is invoked when the user confirms or cancels the dialog. Contains the following properties: <ul style="list-style-type: none"><li>• confirmed - Boolean that indicates if the user has pressed the confirmation button.</li></ul>

## gadgets.sapjam.navigation.registerObjectNavigationHandler

Registers a callback that is invoked when the user navigates to a linked object provided on a gadget activity feed to the gadget.

### Parameters

Name	Description
callback	A function that is invoked when the user navigates to a linked object provided on a gadget activity feed to the gadget. Contains the following properties: <ul style="list-style-type: none"><li>• objectId - The unique identifier of an object to be navigated towards within the gadget.</li></ul>

## gadgets.sapjam.navigation.scrollContainerTo

Allows the gadget to scroll the main window's viewport to a position within the gadget's container (iframe).

### Parameters

Name	Description
leftPosition (integer)	The absolute X position, relative to the gadget's iframe, that the main window should scroll to.
topPosition (integer)	The absolute Y position, relative to the gadget's iframe, that the main window should scroll to.



## **gadgets.sapjam.statusbar.clearBadgeText**

Removes the status bar gadget instance badge if it is being displayed.

## **gadgets.sapjam.statusbar.clearHighlight**

Removes the highlighted state of the status bar gadget instance.

## **gadgets.sapjam.statusbar.hide**

Hides the status bar gadget instance if it is expanded.

## **gadgets.sapjam.statusbar.highlight**

Highlights the status bar gadget instance. A short pulsing animation will be displayed when this function is called.

## **gadgets.sapjam.statusbar.setBadgeText**

Adds a badge beside the status bar gadget instance with some text. This is best for simple information, such as a notification count.

### Parameters

Name	Description
Text	The string to be displayed in the badge.

## **gadgets.sapjam.statusbar.show**

Expands the status bar gadget instance into view if it is hidden.

## 4.1.7.3.6 gadgets.Tab

Provides operations for managing specific tabs within a gadget.

### i Note

To create tabs within a gadget use the `tabset.addTab()` method. To get a gadget's tabs, use the `tabset.getSelectedTab()` or `tabset.getTabs()` methods.

**Required feature:** tabs.

## getCallback

Returns the callback function that is executed when a tab is selected.

### Parameters

Name	Description
------	-------------

none	N.A.
------	------

### Returns

Type	Description
------	-------------

String	The callback function that is executed when a tab is selected.
--------	--

### Example

```
html += "Callback: " + selectedTab.getCallback() + "<br/>";
```

For a more extensive example of the use of the Tab functions, see [Full Tab Code Example \[page 160\]](#).

## getContentContainer

Returns the HTML element where the tab's content is rendered.

### Parameters

Name	Description
------	-------------

none	N.A.
------	------

### Returns

Type	Description
------	-------------

Object	The HTML element where the tab's content is rendered.
--------	---

### Example

```
html += "ContentContainer: " + selectedTab.getContentContainer() + "<br/>";
```

For a more extensive example of the use of the Tab functions, see [Full Tab Code Example \[page 160\]](#).

## getIndex

Returns the tab's index, as a number.

### Parameters

Name	Description
none	N.A.

### Returns

Type	Description
Number	The tab's index number.

### Example

```
html += "TabIndex: " + selectedTab.getIndex() + "<br/>";
```

For a more extensive example of the use of the Tab functions, see [Full Tab Code Example \[page 160\]](#).

## getName

Returns the tab's name, as a string.

### Parameters

Name	Description
none	N.A.

### Returns

Type	Description
String	The tab's name.

### Example

```
html += "TabName: " + selectedTab.getName() + "<br/>";
```

For a more extensive example of the use of the Tab functions, see [Full Tab Code Example \[page 160\]](#).

## getNameContainer

Returns the HTML element that contains the tab's label.

### Parameters

Name	Description
none	N.A.

### Returns

Type	Description
Object	The HTML element that contains the tab's label.

### Example

```
html += "NameContainer: " + selectedTab.getNameContainer();
```

For a more extensive example of the use of the Tab functions, see [Full Tab Code Example \[page 160\]](#).

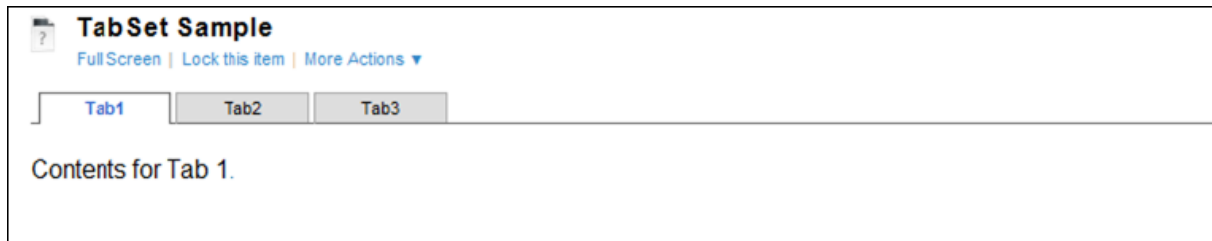
## Full Tab Code Example

This section displays a code example that demonstrates an extensive use of the Tab functions.

```
var tabset=new gadgets.TabSet();
function onLoad(){
    tabset.addTab("Tab1",{contentContainer: document.getElementById("tab1")});
    tabset.addTab("Tab2",{contentContainer: document.getElementById("tab2"),
callback: myCallback});
}
function myCallback(tabId) {
    var p = document.createElement("p");
    var html= "";
    var selectedTab = tabset.getSelectedTab();
    html += "Callback: " + selectedTab.getCallback() + "<br/>";
    html += "ContentContainer: " + selectedTab.getContentContainer() + "<br/>";
    html += "TabIndex: " + selectedTab.getIndex() + "<br/>";
    html += "TabName: " + selectedTab.getName() + "<br/>";
    html += "NameContainer: " + selectedTab.getNameContainer();
    p.innerHTML = html;
    document.getElementById(tabId).appendChild(p);
    gadgets.window.adjustHeight();
}
```

## 4.1.7.3.7 gadgets.Tabset

Provides functionality for adding tabs to, and managing tabs within, a gadget.



**Required feature:** tabs.

### Constructor

Creates a new TabSet object.

#### Parameters

Name	Description
opt_moduleId	( <i>optional</i> ) An optional suffix for the ID of the tab container, as a string..
opt_defaultTab	( <i>optional</i> ) The name of the tab that is selected after the gadget initializes. If not specified, the first tab is selected by default.
opt_container	( <i>optional</i> ) The HTML element to contain the tabs. If not specified, a new <div> element is created and inserted at the top of the gadget.

#### Returns

Type	Description
None	N.A.

#### Example

```
gadgets.TabSet(opt_moduleId, opt_defaultTab, opt_container)
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

### addTab

Adds a new tab within the tabset.

#### Parameters

Name	Description
------	-------------

tabName	The label of the tab, as a string.
opt_params	<i>(optional)</i> An optional object that may contain the following parameters: <ul style="list-style-type: none"> <li>• <b>contentContainer:</b> An existing HTML element to be used as the tab content container. If not specified, a container is created.</li> <li>• <b>callback:</b> A callback function to be executed when the tab is selected.</li> <li>• <b>tooltip:</b> A tooltip description that is displayed when the viewer moves the mouse over the tab.</li> <li>• <b>index:</b> The index at which the insert the tab. If not specified, the tab is appended to the end of the tab set.</li> </ul>

Returns

Type	Description
String	The DOM ID of the tab container.

Example

```
//add a tab
tabset.addTab("Tab1",{contentContainer:document.getElementById("tab1")});
//add a tab with a tooltip
tabset.addTab("Tab2",{contentContainer:document.getElementById("tab2"),
tooltip:"tooltip"});
//insert a tab at the index 1. Specify a callback
tabset.addTab("Tab3",{contentContainer: document.getElementById("tab3"),
callback: myCallback, index:1});
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## alignTabs

Sets the alignment of the tabset within a gadget.

Parameters

Name	Description
align	<i>(optional)</i> Accepts a string. The valid values are left, center, and right. The default value is center.
opt_offset	<i>(optional)</i> The number of pixels to offset tabs from the left or right edge. The default value is 3px.

Returns

Type	Description
None.	N.A.

Example

```
tabset.alignTabs("right");
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## displayTabs

Sets whether or not the tab set will be visible.

### Parameters

Name	Description
display	( <i>optional</i> ) Whether to display or hide the tabs. Accepted values are true and false. The default value is true.

### Returns

Type	Description
None.	N.A.

### Example

```
//display the tabs  
tabset.displayTabs(true);
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## getHeaderContainer

Returns the element that contains the tab headers.

### Parameters

Name	Description
None.	N.A.

### Returns

Type	Description
HTMLElement	The tab header's container element.

### Example

Not currently available.

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## getSelectedTab

Returns the currently selected tab.

### Parameters

Name	Description
None.	N.A.
Returns	
Type	Description
gadgets.Tab	The current selected tab object. For more information on retrieving the properties of the selected tab, see <a href="#">gadgets.Tab [page 158]</a> .
Example	
<pre>var selectedTab = tabset.getSelectedTab();</pre>	

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## getTabs

Returns an array of all the tab objects in the tabset.

Parameters	
Name	Description
None.	N.A.
Returns	
Type	Description
Array.<gadgets.Tab>	An array of existing Tab objects.
Example	
Not currently available.	

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## removeTab

Removes the specified tab and all of its content.

Parameters	
Name	Description
tabIndex	The index of the tab to remove, as a number.
Returns	
Type	Description



None.	N.A.
-------	------

Example

```
//remove the tab at index 3
tabset.removeTab(3);
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## setSelectedTab

Selects the tab at the specified tabIndex and calls the tab's callback function if it exists. If the tab is already selected, the callback function is not called.

Parameters

Name	Description
tabIndex	The index of the tab to select, as a number.

Returns

Type	Description
None.	N.A.

Example

```
//select a tab and execute the callback
tabset.setSelectedTab(2);
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## swapTabs

Swaps the positions of the two specified tabs. The selected tab does not change. No callback functions are called.

Parameters

Name	Description
tabIndex1	The index of the first tab to swap, as a number.
tabIndex2	The index of the second tab to swap, as a number.

Returns

Type	Description
None.	N.A.

## Example

```
//swap the tab with index 1 and 2
tabset.swapTabs(1,2);
```

For an example of an extensive use of the TabSet functions, see [Full TabSet Code Example \[page 166\]](#).

## Full TabSet Code Example

This section displays a code example that demonstrates an extensive use of the TabSet functions.

```
var tabset=new gadgets.TabSet();
function onLoad(){
    //align the tabset to the right of the gadget
    tabset.alignTabs("right");
    //add a tab
    tabset.addTab("Tab1",{contentContainer:document.getElementById("tab1")});
    //add a tab with a tooltip
    tabset.addTab("Tab2",{contentContainer:document.getElementById("tab2"),
tooltip:"tooltip"});
    //insert a tab at the index 1. Specify a callback
    tabset.addTab("Tab3",{contentContainer: document.getElementById("tab3"),
callback: myCallback, index:1});
    //swap the tab with index 1 and 2
    tabset.swapTabs(1,2);
    //display the tabs
    tabset.displayTabs(true);
    //remove the tab at the index 3
    tabset.removeTab(3);
    //select a tab and execute the callback
    tabset.setSelectedTab(2);
}
//display information about the tab when it is selected.
function myCallback {
    var p = document.createElement("p");
    var selectedTab = tabset.getSelectedTab();
    var html += "TabName: " + selectedTab.getName();
    p.innerHTML = html;
    document.getElementById(tabId).appendChild(p);
    gadgets.window.adjustHeight();
}
gadgets.util.registerOnLoadHandler(onLoad);
```

### 4.1.7.3.8 gadgets.util

Provides general-purpose utility functions, such as escaping and unescaping strings, sanitizing HTML, and registering onload handlers.

**Required feature:** None

## gadgets.util.escapeString

Escapes the provided input using HTML entities, to make the input safer.

Parameters	
Name	Description
String	The string to escape.
Returns	
Type	Description
String	The escaped string.
Example	
See final function, <code>gadgets.util.unescapeString</code> .	

## gadgets.util.getFeatureParameters

Returns the values of the parameters for the specified feature.

Parameters	
Name	Description
feature	The feature to get parameters for, as a string.
Returns	
Type	Description
Object	An object containing the parameters for the feature, or null.
Example	
See final function, <code>gadgets.util.unescapeString</code> .	

## gadgets.util.hasFeature

Returns whether the specified feature is supported.

Parameters	
Name	Description
feature	The feature to get parameters for, as a string.
Returns	
Type	Description
Boolean	True if the feature is supported, otherwise false.

#### Example

See final function, `gadgets.util.unescapeString`.

## **gadgets.util.registerOnLoadHandler**

Registers an onload handler, a function that is executed when the gadget loads. Multiple handlers can be registered. Handlers are invoked in the same order in which they are registered.

#### Parameters

Name	Description
callback	The callback function to run.

#### Returns

Type	Description
none	N.A.

#### Example

See final function, `gadgets.util.unescapeString`.

## **gadgets.util.sanitizeHtml**

Sanitizes the specified string. The returned value may include HTML tags. To get plain text, use `gadgets.util.escapeString` instead.

#### Parameters

Name	Description
String	The string to sanitize.

#### Returns

Type	Description
String	The sanitized version of the specified string.

#### Example

See final function, `gadgets.util.unescapeString`.

## **gadgets.util.unescapeString**

Reverses `gadgets.util.escapeString`.

#### Parameters

Name	Description
String	The string to unescape.

#### Returns

Type	Description
String	The unescaped string.

#### Example

```
function utilTest(){
    var html = "The escaped string is: " + gadgets.util.unescapeString("Hello
<> Jason!");
    html += "<br/> has setprefs?: " + gadgets.util.hasFeature("setprefs") + ".";
    html += "<br/> getFeatureParameters for 'setprefs': " +
gadgets.util.getFeatureParameters("setprefs") + ".";
    document.getElementById("content_div").innerHTML = html;
}
gadgets.util.registerOnLoadHandler(utilTest);
```

### 4.1.7.3.9 gadgets.views

Provides the ability to run different code depending on the "view" value, which can be set to "home" (for Content gadgets) or "profile" (for Profile gadgets).

**Required feature:** None

Currently SAP Jam Collaboration supports "home", "preview", and "profile" views.

`gadgets.views.requestNavigateTo` is not supported and currently we cannot switch between different views once the gadget is loaded. At the moment use "profile" view for profile gadgets and "home" view for content gadgets. Example:

```
<Content type="html" view="home">
  <![CDATA[
    <p> We are in home (content) view! </p>
  ]]>
</Content>
<Content type="html" view="profile">
  <![CDATA[
    <p> We are in profile (profile) view! </p>
  ]]>
</Content>
</Module>
```

The above gadget's content instance will render "We are in home (content) view!" and profile instance will render "We are in profile (profile) view!". Note that an administrator still needs to register the gadget twice: once with 'content' context and once with 'profile' context.

## gadgets.views.openGadget

Opens a gadget in the container UI. The location of the gadget site in the container will be determined by the view target passed in.

### Parameters

Name	Description
resultCallback	<b>Type:</b> {Function}  Called by the gadget when the gadget closes. This function will be called with the return value as a parameter.
navigateCallback	<b>Type:</b> {Function}  Called by the gadget with the site handler and metadata as parameters. The site handler can only be used as a parameter in gadgets.views.close(opt_site_handler).
opt_params	<b>Type:</b> {Object}  These are optional parameters which can be used to open gadgets. The following parameters may be included in this object: <ul style="list-style-type: none"><li>• <b>view</b> {String}: The surface view that indicates the type of gadget site.</li><li>• <b>viewTarget</b> {String}: The view that indicates where to open the gadget. Supported "view targets" are: MODALDIALOG, DIALOG (modeless) and FLOAT (modeless).</li><li>• <b>viewParams</b> {Object}: View parameters for the view being rendered.</li><li>• <b>coordinates</b> {Object}: Object containing the desired absolute positioning css parameters (top bottom left right). All values are relative to the calling gadget.</li></ul>
Usage	<static> gadgets.views.openGadget(resultCallback, navigateCallback, opt_params)

### Example 1 - Opening and closing a dialog:

Step 1 - Add the following gadget spec XML:

```
<Content type="html" href="helloworld.html"/>
<Content type="html" href="dialog.html" view="hw_dialog"/>
```

Step 2 - Use the following Javascript to create the dialog box:

```
gadgets.views.openGadget(function(result) {
    // Handle the result from the dialog box...
}, function(site) {}, {
    view: 'hw_dialog',
    viewTarget: 'MODALDIALOG'
});
```

Step 3 - Use the following Javascript in the dialog box, to return a value and close itself (e.g. after clicking on an OK button):

```
gadgets.views.setReturnValue(result);
gadgets.views.close();
```

### Example 2 - Passing large data be passed between gadget views (In a view launching a dialog):

```
gadgets.sapjam.asyncMessage.post(someBigObject, function(token) {
  gadgets.views.openGadget(
    function(result) { ... },
    function(site) { ... },
    {
      view: 'some-view',
      viewTarget: 'MODALDIALOG',
      viewParams: {paramsKey: token}
    }
  );
});
```

**Example 3 - Passing large data be passed between gadget views (In a dialog launched by another view):**

```
gadgets.util.registerOnLoadHandler(function() {
  var params = gadgets.views.getParams();
  gadgets.sapjam.asyncMessage.get(params.paramsKey, function(someBigObject) {
    // use someBigObject...
  });
});
```

### 4.1.7.3.10 gadgets.window

Provides operations for getting information about and modifying the window the gadget is placed in.

**Required features:** settitle, dynamic-height.

#### gadgets.window.adjustHeight

Adjusts the height of the gadget.

Parameters

Name	Description
opt_height	The preferred height of the gadget, in pixels. If not specified, will attempt to fit the gadget to its content.

Returns

Type	Description
None.	N.A.

Example

```
gadgets.window.adjustHeight();
```

## gadgets.window.getViewPortDimensions

Returns an object with height and width properties for the gadget.

### Parameters

Name	Description
None.	N.A.

### Returns

Type	Description
Object	An object with width and height properties.

### Example

```
function get_dimensions() {
    var dimensions = gadgets.window.getViewPortDimensions();
    var html = "Height: " + dimensions.height + " Width: " + dimensions.width;
    document.getElementById("content_div").innerHTML = html;
}
```

## gadgets.window.setTitle

Sets the title of the gadget.

### Parameters

Name	Description
title	A string representing the title of the gadget.

### Returns

Type	Description
None.	N.A.

### Example

```
function changeTitle(form) {
    var newTitle = form.inputbox.value;
    gadgets.window.setTitle(newTitle);
}
```

## 4.1.7.4 OpenSocial API (osapi) reference

This API reference documents the SAP Jam Collaboration-supported parts of osapi API.



## 4.1.7.4.1 osapi.appdata

Provides functionality for retrieving, updating, and deleting a user's data.

**Required feature:** osapi

### osapi.appdata.get

Retrieves the appdata of the participants of the Activity that the gadget has been added to.

#### Parameters

Name	Description
userId	( <i>optional</i> ) The id of the user whose data you want to retrieve. Either the SAP Jam ID of the user, or '@viewer', to retrieve the data of the gadget viewer.
groupId	( <i>optional</i> ) The id of the group whose data you want to retrieve.
keys	The key or keys that you want to retrieve.

#### Returns

Type	Description
Object	A request object which, when executed, performs the get request.

#### Example

```
var appdata_get = function() {
    osapi.appdata.get({userId: '@viewer', groupId: '@self', keys:
    ['stuff']}).execute(function(userData) {
        document.getElementById("osapi").innerHTML += "<br>osapi.appdata.get: "
    + gadgets.json.stringify(data) + ".";
    });
};
```

### osapi.appdata.update

Creates or updates the appdata of the viewer of the Activity. `osapi.appdata.update` cannot be called by the viewer to update the data of another Activity participant.

#### Parameters

Name	Description
------	-------------

<b>data</b>	A JSON-formatted object containing the key/value pairs to be created or updated.
Returns	
Type	Description
<b>Object</b>	A request object which, when executed, performs the update request.
Example	
<pre>var appdata_update = function () {     var input = document.getElementById('osapi').value;     osapi.appdata.update({userId: '@viewer', groupId: '@self', data: {stuff: input}}).execute(function (userData) {         if (userData.error) {             alert(userData.error.message)         }         else {             alert('The data has been updated: ' + input);         }     }); };</pre>	

## osapi.appdata.delete

Deletes the appdata of the viewer of the Activity. `osapi.appdata.delete` cannot be called by the viewer to delete the data of another Activity participant.

Parameters	
Name	Description
<b>keys</b>	An array of keys that identify the data values to delete.
Returns	
Type	Description
<b>Object</b>	A request object which, when executed, performs the delete request.
Example	
<pre>var appdata_update = function () {     var input = document.getElementById('osapi').value;     osapi.appdata.delete({keys : ['stuff']}).execute(function (userData) {         if (userData.error) {             alert(userData.error.message)         }         else {             alert('The data has been deleted');         }     }); };</pre>	

## 4.1.7.4.2 osapi.people

The `osapi.people` API calls retrieve Person information about specified members, viewers, viewer's friends, owners, and owner's friends.

**Required features:** `osapi`

For all `osapi.people` API calls, there is a single parameter, which is a JavaScript object representing a GET method for retrieving [Person \[page 179\]](#) information. The information requested by the object varies for each API call.

All API calls return a request object which, when executed, performs the indicated GET request and retrieves the requested or default Person information, or a collection of such information.

The following API calls are supported for both content and profile gadgets, although the content returned differs.

### i Note

SAP Jam Collaboration does not support multiple userIds (targets) with the following groupIds (circles): @friends, @all. Returns 400.

## osapi.people.get

Builds a request to retrieve information for each [Person \[page 179\]](#) specified in the parameter object. When no parameter is specified, the request will return the default information (@viewer + @self).

### Parameters

Name	Description
N.A.	A JavaScript object of the user or users requested for the <a href="#">Content Gadgets request parameters [page 178]</a> or <a href="#">Profile Gadgets request parameters [page 178]</a> . If no parameter is specified, the request will return the default information (@viewer + @self).

### Returns

Type	Description
Object or Objects	An object containing Member information or a collection of such objects.

### Example

```
osapi.people.get({
  "userId": "UUID_HERE"
}).execute(function(data) {
  document.getElementById("osapi").innerHTML += "<br>osapi.people.get(): " +
  data.displayName + "<br>";
});
```

## osapi.people.getViewer

**Profile & Content:** Returns [Person \[page 179\]](#) information about the user currently viewing the SAP Jam osapi content or profile gadget. When no parameter is specified, the request will return the default information (@viewer + @self).

### Parameters

Name	Description
N.A.	A JavaScript object of the user or users requested for the <a href="#">Content Gadgets request parameters [page 178]</a> or <a href="#">Profile Gadgets request parameters [page 178]</a> , except that the value for the userId parameter is set to '@viewer'. If no parameter is specified, the request will return the default information (@viewer + @self).

### Returns

Object	An object containing Member information about the viewer.
--------	---

### Example

```
osapi.people.getViewer().execute(function(data) {  
    document.getElementById("osapi").innerHTML += "<br>osapi.people.getViewer:"  
    " + data.displayName + "<br>";  
});
```

## osapi.people.getViewerFriends

**Profile:** Returns a collection of information about each [Person \[page 179\]](#) that the viewer is following. When no parameter is specified, the request will return the default information (@viewer + @friends).

**Content:** Returns a collection of information about each [Person \[page 179\]](#) in the group in which the osapi content gadget has been added, including the viewer. When no parameter is specified, the request will return the default information (@viewer + @friends).

### Parameters

Name	Description
N.A.	A JavaScript object of the user or users requested for the <a href="#">Content Gadgets request parameters [page 178]</a> or <a href="#">Profile Gadgets request parameters [page 178]</a> , except that the value for the userId parameter is set to '@viewer' and the groupId parameter is set to '@friends'. If no parameter is specified, the request will return the default information (@viewer + @friends).

### Returns

Type	Description
Objects	A collection of objects containing Member information about the viewer's friends.

### Example

```
osapi.people.getViewerFriends().execute(function(data) {
    console.log(data);
    document.getElementById("osapi").innerHTML +=
"osapi.people.getViewerFriends: ";
    for (i = 0; i < data.totalResults; i++) {
        document.getElementById("osapi").innerHTML += data.list[i].displayName
+ " ";
    }
    document.getElementById("osapi").innerHTML += "<br>";
});
```

## osapi.people.getOwner

**Profile:** Returns information about the [Person \[page 179\]](#) whose profile is hosting the gadget.

**Content:** Returns information about the [Person \[page 179\]](#) who is the owner of the content gadget. When no parameter is specified, the request will return the default information (@owner + @self).

### Parameters

Name	Description
N.A.	A JavaScript object of the user or users requested for the <a href="#">Content Gadgets request parameters [page 178]</a> or <a href="#">Profile Gadgets request parameters [page 178]</a> , except that the value for the userId parameter is set to '@owner'. If no parameter is specified, the request will return the default information (@owner + @self).

### Returns

Type	Description
Object	An object containing Member information about the owner.

### Example

```
osapi.people.getOwner().execute(function(data) {
    document.getElementById("osapi").innerHTML += "<br>osapi.people.getOwner: "
+ data.displayName + "<br>";
});
```

## osapi.people.getOwnerFriends

**Profile:** Returns information about each [Person \[page 179\]](#) that the owner is following. When no parameter is specified, the request will return the default information (@owner + @friends).

**Content:** Returns information about each [Person \[page 179\]](#) that is a member of the group in which the osapi content gadget has been added, including the owner. When no parameter is specified, the request will return the default information (@owner + @friends).

### Parameters

Name	Description
N.A.	A JavaScript object of the user or users requested for the <a href="#">Content Gadgets request parameters [page 178]</a> or <a href="#">Profile Gadgets request parameters [page 178]</a> , except that the value for the <code>userId</code> parameter is set to '@owner' and the <code>groupId</code> parameter is set to '@friends'. If no parameter is specified, the request will return the default information (@owner + @friends).
Returns	
Type	Description
Objects	A collection of objects containing Member information about the owner's friends.
Example	
<pre>osapi.people.getOwnerFriends().execute(function(data) {     document.getElementById("osapi").innerHTML +=     "osapi.people.getOwnerFriends: ";     for (i = 0; i &lt; data.totalResults; i++) {         document.getElementById("osapi").innerHTML += data.list[i].displayName     + " ";     }     document.getElementById("osapi").innerHTML += "&lt;br&gt;"; });</pre>	

## Content Gadgets request parameters

Content Gadgets	@self	@friends	@all
@me/@viewer	logged in user	all members of the group	same as @friends
@owner	owner of the gadget	all members of the group	same as @friends
@uuid	user by uuid	all members of the group	same as @friends

## Profile Gadgets request parameters

Content Gadgets	@self	@friends	@all
@me/@viewer	logged in user	people logged in user is following	same as @friends
@owner	user who owns the profile	people profile owner is following	same as @friends
@uuid	user by uuid	people that user with specified uuid is following	same as @friends

## Returned Person information

The preceding `osapi.people` API calls return the following fields of information for each person:

- `id` (uuid)
- `displayName`
- `name.givenName` (firstname)
- `name.familyName` (lastname)
- `name.formatted` (firstname and lastname)
- `emails[#].type` (email type - home, work, etc.)
- `emails[#].value` (email address)
- `thumbnailUrl` (**absolute** thumbnail URL)
- `photos[#].type` (photo type - thumbnail, etc.)
- `photos[#].value` (**absolute** photo URL)

## Full Tab Code Example

This section displays a code example that demonstrates an extensive use of the `osapi.people` functions.

```
// Calls and displays all endpoints for all osapi.people API calls:
// - osapi.people.getViewer()
// - osapi.people.getOwner()
// - osapi.people.getViewerFriends()
// - osapi.people.getOwnerFriends()
// - osapi.people.get()
function makeOSAPIpeopleCall() {

    // Content and Profile Gadgets:
    // - Creates an object (data) that contains Person information about the
    user currently
    // viewing the gadget
    osapi.people.getViewer().execute(function(data) {

        // Shows the "data" object in the console
        console.log(data);
        // Creates HTML that shows all osapi.people.getViewer endpoints for the
        "data" object
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
        "<p>-----</p>";
        osapiOutput += "<p><h2>osapi.people.getViewer</h2></p>";
        osapiOutput += "<ul>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>displayName</b> = " + data.displayName + "</li>";
        osapiOutput += "<li>data.<b>name.givenName</b> = " + data.name.givenName
        + "</li>";
        osapiOutput += "<li>data.<b>name.familyName</b> = " +
        data.name.familyName + "</li>";
        osapiOutput += "<li>data.<b>name.formatted</b> = " + data.name.formatted
        + "</li>";
        osapiOutput += "<li>data.<b>emails[0].type</b> = " + data.emails[0].type
        + "</li>";
        osapiOutput += "<li>data.<b>emails[0].value</b> = " +
        data.emails[0].value + "</li>";
```

```

        osapiOutput += "<li>data.<b>thumbnailUrl</b> = " + data.thumbnailUrl +
"</li>";
        osapiOutput += "<li>data.<b>photos[0].type</b> = " + data.photos[0].type
+ "</li>";
        osapiOutput += "<li>data.<b>photos[0].value</b> = " +
data.photos[0].value + "</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
"<p>-----</p>";
        osapiOutput += "<p></p>";
        // Appends HTML to the <body> element
        $("body").append(osapiOutput);
    });
    // Content Gadgets:
    // - Creates an object (data) that contains Person information about the
user who is
    //   the owner of the gadget
    // Profile Gadgets:
    // - Creates an object (data) that contains Person information about the
user whose
    //   profile is hosting the gadget
    osapi.people.getOwner().execute(function(data) {

        // Shows the "data" object in the console
        console.log(data);
        // Creates HTML that shows all osapi.people.getOwner endpoints for the
"data" object
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
"<p>-----</p>";
        osapiOutput += "<p><h2>osapi.people.getOwner</h2></p>";
        osapiOutput += "<ul>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>displayName</b> = " + data.displayName + "</li>";
        osapiOutput += "<li>data.<b>name.givenName</b> = " + data.name.givenName
+ "</li>";
        osapiOutput += "<li>data.<b>name.familyName</b> = " +
data.name.familyName + "</li>";
        osapiOutput += "<li>data.<b>name.formatted</b> = " + data.name.formatted
+ "</li>";
        osapiOutput += "<li>data.<b>emails[0].type</b> = " + data.emails[0].type
+ "</li>";
        osapiOutput += "<li>data.<b>emails[0].value</b> = " +
data.emails[0].value + "</li>";
        osapiOutput += "<li>data.<b>thumbnailUrl</b> = " + data.thumbnailUrl +
"</li>";
        osapiOutput += "<li>data.<b>photos[0].type</b> = " + data.photos[0].type
+ "</li>";
        osapiOutput += "<li>data.<b>photos[0].value</b> = " +
data.photos[0].value + "</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
"<p>-----</p>";
        osapiOutput += "<p></p>";
        // Appends HTML to the <body> element
        $("body").append(osapiOutput);
    });
    // Content Gadgets:
    // - Creates a collection object (data) that contains Person information
about each
    //   Person in the group in which the gadget has been added, including the
viewer
    // Profile Gadgets:

```



```

    // - Creates a collection object (data) that contains Person information
    about each
    //    Person that the viewer is following
    osapi.people.getViewerFriends().execute(function(data) {

        // Shows the "data" object in the console
        console.log(data);
        // Creates HTML
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
"<p>-----</
p>";

        osapiOutput += "<p><h2>osapi.people.getViewerFriends:</h2></p>";
        osapiOutput += "<ul>";
        // Creates HTML that shows all osapi.people.getViewerFriends endpoints
    for each
        // person in the "data" object
        for (i = 0; i < data.list.length; i++) {
            osapiOutput += "<li>data.list[" + i + "].<b>id</b> = " +
data.list[i].id + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>displayName</b> = " +
data.list[i].displayName + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>givenName</b> = " +
data.list[i].name.givenName + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>familyName</b> = " +
data.list[i].name.familyName + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>formatted</b> = " +
data.list[i].name.formatted + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>emails[0].type</b> = " +
data.list[i].emails[0].type + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>emails[0].value</b> = " +
+ data.list[i].emails[0].value + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>thumbnailUrl</b> = " +
data.list[i].thumbnailUrl + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>photos[0].type</b> = " +
data.list[i].photos[0].type + "</li>";
            osapiOutput += "<li>data.list[" + i + "].<b>photos[0].value</b> = " +
+ data.list[i].photos[0].value + "</li>";
        }
        osapiOutput += "</ul>";
        osapiOutput +=
"<p>-----</
p>";

        osapiOutput += "<p></p>";
        // Appends HTML to the <body> element
        $("body").append(osapiOutput);
    });
    // Content Gadgets:
    // - Creates a collection object (data) that contains Person information
    about each Person
    //    that is a member of the group in which the gadget has been added,
    including the owner
    // Profile Gadgets:
    // - Creates a collection object (data) that contains Person information
    about each Person
    //    that the owner is following
    osapi.people.getOwnerFriends().execute(function(data) {
        // Shows the object (data) in the console
        console.log(data);
        // Creates HTML that shows all osapi.people.getOwnerFriends endpoints
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
"<p>-----</
p>";

        osapiOutput += "<p><h2>osapi.people.getOwnerFriends:</h2></p>";
        osapiOutput += "<ul>";

```

```

        // Creates HTML that shows all osapi.people.getOwnerFriends endpoints
for each
    // person in the "data" object
    for (i = 0; i < data.list.length; i++) {
        osapiOutput += "<li>data.list[" + i + "].<b>id</b> = " +
data.list[i].id + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>displayName</b> = " +
data.list[i].displayName + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>givenName</b> = " +
data.list[i].name.givenName + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>familyName</b> = " +
data.list[i].name.familyName + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>formatted</b> = " +
data.list[i].name.formatted + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>emails[0].type</b> = " +
data.list[i].emails[0].type + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>emails[0].value</b> = " +
data.list[i].emails[0].value + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>thumbnailUrl</b> = " +
data.list[i].thumbnailUrl + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>photos[0].type</b> = " +
data.list[i].photos[0].type + "</li>";
        osapiOutput += "<li>data.list[" + i + "].<b>photos[0].value</b> = " +
data.list[i].photos[0].value + "</li>";
    }
    osapiOutput += "</ul>";
    osapiOutput +=
"<p>-----</p>";
    osapiOutput += "<p></p>";
    // Appends HTML to the <body> element
    $("body").append(osapiOutput);
});
// Content and Profile Gadgets:
// - Creates an object (dataForID) that contains Person information about
the user currently
//   viewing this gadget
osapi.people.getViewer().execute(function(dataForID) {

    // Shows the "dataForID" object in the console
    console.log(dataForID);
    // Creates an object (data) that contains Person information of the user
specified by
    // the "dataForID" object (dataForID.id)
    osapi.people.get({"userId": dataForID.id}).execute(function(data) {

        // Shows the "data" object in the console
        console.log(data);
        // Creates HTML that shows all osapi.people.get endpoints for the
"data" object
        var osapiOutput = "";
        osapiOutput += "<p></p>";
        osapiOutput +=
"<p>-----</p>";
        osapiOutput += "<p><h2>osapi.people.get</h2></p>";
        osapiOutput += "<p><b>using the id from osapi.people.getViewer</b></p>";
        osapiOutput += "<p><b>osapi.people.getViewer.dataForID.<b>id</b> = " +
dataForID.id + "</b></p>";
        osapiOutput += "<ul>";
        osapiOutput += "<li>data.<b>id</b> = " + data.id + "</li>";
        osapiOutput += "<li>data.<b>displayName</b> = " + data.displayName +
"</li>";
        osapiOutput += "<li>data.<b>name.givenName</b> = " +
data.name.givenName + "</li>";
        osapiOutput += "<li>data.<b>name.familyName</b> = " +
data.name.familyName + "</li>";
    });
}

```

```

        osapiOutput += "<li>data.<b>name.formatted</b> = " +
data.name.formatted + "</li>";
        osapiOutput += "<li>data.<b>emails[0].type</b> = " +
data.emails[0].type + "</li>";
        osapiOutput += "<li>data.<b>emails[0].value</b> = " +
data.emails[0].value + "</li>";
        osapiOutput += "<li>data.<b>thumbnailUrl</b> = " + data.thumbnailUrl
+ "</li>";
        osapiOutput += "<li>data.<b>photos[0].type</b> = " +
data.photos[0].type + "</li>";
        osapiOutput += "<li>data.<b>photos[0].value</b> = " +
data.photos[0].value + "</li>";
        osapiOutput += "</ul>";
        osapiOutput +=
"<p>-----</p>";
        osapiOutput += "<p></p>";
        // Appends HTML to the <body> element
$("body").append(osapiOutput);
    });
}

```

## Download the Gadget Sample Code

1. Download the gadget source code from our official SAP Github repository at:  
<https://github.com/SAP/SAPJamSampleCode> 📄
2. Extract the file.
3. Upload the "\\OpenSocial\\Gadget\\OSAPI\_people" folder to your webhost.

## RUN THE GADGET!

Register the URL of OSAPI\_people.xml with SAP Jam, and run it.

If you do not have a publicly accessible server to host from, register the prebuilt version of our gadget with the following URL, and run it:

[https://SAPJamSampleCode.github.io/OpenSocial/Gadget/OSAPI\\_people/OSAPI\\_people.xml](https://SAPJamSampleCode.github.io/OpenSocial/Gadget/OSAPI_people/OSAPI_people.xml) 📄

Congratulations! The gadget should be up and running!

### Troubleshooting

If you are having any problems registering our gadget with your SAP Jam live service, see:

- [Register your gadget with SAP Jam Collaboration \[page 74\]](#)

## References

<http://opensocial-resources.googlecode.com/svn/spec/2.5/Social-Gadget.xml#osapi.people> 📄

<http://opensocial-resources.googlecode.com/svn/spec/2.5/Social-API-Server.xml#rfc.section.2.1.1.2.2> 

## 4.1.7.5 Wave API reference

This API reference documents the SAP Jam Collaboration-supported parts of wave API, which provides AJAX dynamic updates to gadget data.


The Wave API originated as a Google Development project, but it is currently transitioning to being an Apache Incubator project. Access to the Wave documentation is therefore currently a bit scattered and will likely change in the near future.

- See the [Google Wave Gadgets API](#)  documentation.
- Visit the [Apache Incubator Wave](#)  project site.

### 4.1.7.5.1 wave.Participants

The Wave Participants API allows OpenSocial gadgets to get "soft real-time" (polling speed depends on whether the gadget is in focus) updates for users' content. The polling is made efficient by appropriate use of caching, both browser-side and server-side.

#### setStateCallback(callback, opt\_context)

Uses the passed `callback` parameter to define a callback function to be run when the gadget's shared state object changes. You can optionally specify an object (`opt_context`) to receive the callback. The `setStateCallback()` callback function is always called once when the state is first received by the gadget, and then subsequently whenever a new state is received by the gadget. See the section [Structuring a Wave Gadget](#)  for more discussion of how to use `setStateCallback()`.

Defines a callback function to be run when the gadget's shared state object changes.

##### Parameters

Name	Description
<code>callback</code>	Defines a callback function to be run when the gadget's shared state object changes.
<code>opt_context</code>	( <i>optional</i> ) Can be used to set an object to receive the callback.

##### Returns

Type	Description
None.	N.A.

##### Example

```
function init() {
    if (wave && wave.isInWaveContainer()) {
        wave.setStateCallback(stateUpdated);
    }
}
gadgets.util.registerOnLoadHandler(init);
```

## setPrivateStateCallback(callback, opt\_context)

Uses the passed `callback` parameter to define a callback function to be run when the gadget's private state object changes. You can optionally specify an object (`opt_context`) to receive the callback. The `setPrivateStateCallback()` callback function is always called once when the private state is first received by the gadget, and then subsequently whenever a new private state is received by the gadget.

### Parameters

Name	Description
<code>callback</code>	Defines a callback function to be run when the gadget's private state object changes.
<code>opt_context</code>	( <i>optional</i> ) Can be used to set an object to receive the callback.

### Returns

Type	Description
None.	N.A.

### Example

```
function init() {
    if (wave && wave.isInWaveContainer()) {
        wave.setPrivateStateCallback(stateUpdated);
    }
}
gadgets.util.registerOnLoadHandler(init);
```

## setParticipantCallback(callback, opt\_context)

Uses the passed `callback` parameter to define a callback function to be run when there is a change to the participants of the wave in which the gadget resides. This includes participants getting loaded when the gadget first runs, participants joining or leaving the wave, or changes to a participant's information, such as a name change. This does not include a participant performing an operation within the wave.

You can optionally specify an object (`opt_context`) to receive the callback.

### Parameters

Name	Description
callback	Defines a callback function to be run when there is a change to the participants of the wave in which the gadget resides.
opt_context	( <i>optional</i> ) Can be used to set an object to receive the callback.
Returns	
Type	Description
None.	N.A.
Example	
codeExample	

## getState()

Returns the gadget's shared state object, which conceptually is a key-value map. Once you have the state object, you can perform operations on it like querying for the value of particular keys. For example, `wave.getState().get('count')` returns the value for the `count` key. Note that both keys and values must be strings.

Parameters	
Name	Description
None.	N.A.
Returns	
Type	Description
Object	Returns the gadget's shared state object, which conceptually is a key-value map.
Example	
<pre>function buttonClicked() {     var value = parseInt(wave.getState().get('count', '0'));     wave.getState().submitDelta({'count': value + 1}); }  function stateUpdated() {     if(!wave.getState().get('count')) {         div.innerHTML = "The count is 0."     }     else {         div.innerHTML = "The count is " + wave.getState().get('count');     } }</pre>	

## getPrivateState()

Returns the gadget's private state object, which is similar to the shared state object, but contains key-value pairs for the private gadget state.

### Parameters

Name	Description
None.	N.A.

### Returns

Type	Description
Object	Returns the gadget's private state object, which is similar to the shared state object, but contains key-value pairs for the private gadget state.

### Example

```
codeExample
```

## submitDelta(delta)

Updates the state object with `delta`, which is a map of key-value pairs representing an update. For example, `wave.getState().submitDelta({'count': 5})` or `wave.getPrivateState().submitDelta({'count': 5})` sets the value of the `count` key to '5' in respectively shared or private state. Note that all participants will see the update in the shared state, but the update in private state will only be visible to the viewer.

### Parameters

Name	Description
<code>delta</code>	A map of key-value pairs representing an update.

### Returns

Type	Description
None.	N.A.

### Example

```
// Reset value of "count" to 0
function resetCounter() {
  wave.getState().submitDelta({'count': '0'});
}
```

## Full wave.Participants Code Example

This section displays a code example that demonstrates an extensive use of the Tab functions.

```
<?xml version="1.0" encoding="UTF-8" ?>
<Module specificationVersion='2'>
  <ModulePrefs title="State Example" height="120">
    <Require feature="wave"/>
  </ModulePrefs>
  <Content type="html">
    <![CDATA[
<div id="content_div" style="height: 50px;"></div>
<script type="text/javascript">

  var div = document.getElementById('content_div');

  function buttonClicked() {
    var value = parseInt(wave.getState().get('count', '0'));
    wave.getState().submitDelta({'count': value + 1});
  }

  function stateUpdated() {
    if(!wave.getState().get('count')) {
      div.innerHTML = "The count is 0."
    }
    else {
      div.innerHTML = "The count is " + wave.getState().get('count');
    }
  }

  function init() {
    if (wave && wave.isInWaveContainer()) {
      wave.setStateCallback(stateUpdated);
    }
  }
  gadgets.util.registerOnLoadHandler(init);

  // Reset value of "count" to 0
  function resetCounter(){
    wave.getState().submitDelta({'count': '0'});
  }

</script>
<input type=button value="Click Me!" id="butCount" onClick="buttonClicked()">
<input type=button value="Reset" id="butReset" onClick="resetCounter()">
]]>
</Content>
</Module>
```

## References

See <http://www.waveprotocol.org/wave-apis/google-wave-gadgets-api> ➡



# 5 Integrate Webhooks

## 5.1 Push notifications for webhooks

Webhooks provide a means of tracking specific events that occur in SAP Jam and then sending the event notification metadata to third party applications that use them. From the SAP Jam Admin console, company administrators can create and manage webhook subscriptions.

For full documentation on the Webhooks API, see the [SAP Jam Webhooks API Reference](#).

The following tutorials will show you how to integrate webhooks into your applications.

### 5.1.1 Webhooks - Alias Users

Webhooks allow your application to subscribe to certain changes on Jam and receive real time updates.

Whenever a subscribed change has happened, Jam will make a HTTP POST to the callback URL you specify. This eliminates the need for continuous requests on our OData API to check for updates, saving you significant network bandwidth and computation costs.

### Get Started with Webhooks

We have a sample Java EE servlet to demonstrate how webhooks on Jam should be used. This guide will show you how to run the sample code on SAP BTP. If you don't have a SAP BTP account, follow [this tutorial](#) to set one up.

You will also need Eclipse (Java EE Edition) with SAP BTP Tools Plugin installed. Follow [this guide](#) if you don't have that set up.

### Download the Sample Webhooks Servlet

The following steps will demonstrate how to setup the sample webhooks servlet.

1. Download the sample webhooks servlet source code from our [official SAP Github repository](#).
2. Extract the zip file.

## Import the JamSampleWebhookClient Project

1. Open Eclipse.
2. Right click in the **Project Explorer** panel and select **Import > Import...**
3. Select **Maven > Existing Maven Projects**.
4. Click **Next**.
5. Click **Browse...**
6. Navigate to **/Webhooks/Java/JamWebHooksTester4j** and click **Open**.
7. Click **Finish**.

## Deploy the Servlet

Although the servlet is not ready yet, we want to deploy it first so we can get a public URL to set up a webhook push notification subscription on Jam.

1. In the **Project Explorer** expand the tree as follows: **SAPJamSampleWebhooksServer > Java Resources > src > com.sap.jam.webhooks.sample**
2. Right click on **WebhookServlet.java** and select **Run As > Run on Server**.
3. Select **Manually define a new server**.
4. Choose **SAP > SAP BTP** as the server type.
5. Enter **hanatrial.ondemand.com** as your Landscape host (or **hana.ondemand.com** if you are using a non-trial account).
6. Click **Next**.
7. Enter your application name in the **Application name** field.
8. Enter your SAP BTP account information in the **Account information** fields.
9. Click **Finish**.

Once deployment is done, your browser should automatically open a link to the newly deployed server with a basic message. Take note of the URL.

## Create an Alias User on Jam

We want to trigger a callback whenever an Alias user is @mentioned. Let's create an Alias user first:

1. In SAP Jam, click on the "cog" settings icon and select **Admin**. The SAP Jam Admin page will appear.
2. Select **Users > Alias Accounts**.
3. Click **New Alias Account**.
4. Create an Alias Account Name in the **Alias Account Name** field.
5. Click **Add an OAuth2 Access Token**. If you do not have an OAuth2 Access Token [click here \[page 14\]](#) to learn how.
6. Select an **OAuth Client** from the drop down menu and click **OK**.
7. Click **Save changes**.
8. Copy the new **OAuth2 Access Token** and paste it as the **JAM\_OAUTH\_TOKEN** static string in **WebhookServlet.java**.

**Manage Alias Account**

Basic Profile Information

Alias UserID: iG51Kp6JiLHXA9P30frdcc

Alias Account Name: Webhook Bot

Description:

Alias Account Name

API Access

OAuth2 Access Token: [Masked Token]

For OAuth Client: Webhooks

Delete OAuth2 Access Token

OAuth2 Access Token

## Create a new Push Notification Subscription on Jam

With an alias user created, we could create a new push notification subscription that triggers webhook calls whenever the alias user is @mentioned.

1. In the admin settings page select **Integrations > Push Notifications**.
2. Click **+Push Notification Subscription**.
3. Create a name for the Push Notification in the **Name** field.
4. Paste the URL of the Java Servlet you just deployed on SAP BTP ([https://Servlet\\_URL/](https://Servlet_URL/)) into the **Callback URL** field.
5. Set the **Scope** to **Alias Account** and enter the alias user we just created in the **Alias Account** field.
6. Enable all the **Alias Account Notifications** in **Event List**.
7. Copy the **Token** field and paste it as the `JAM_WEBHOOK_VERIFICATION_TOKEN` static string in `WebhookServlet.java`.
8. Click **Save**.

**Manage Push Notification Subscription**

Subscription Type: Webhook ☐ Enabled

Name\*: Sample Alias Subscription

Description:

Name

Callback, Token, Scope, Event List

## Redeploy Your Servlet on SAP BTP

You should now have the OAuth and verification token properly configured in the servlet code. Let's redeploy your servlet on SAP BTP:

1. Select the **Servers** tab.
2. Right click on your **SAP BTP server** and select **Publish**.

Your server should be now redeployed with the new changes.

## Enable Your Webhook in Jam

Let's enable your webhook in Jam:

1. Now go back to Jam and select **Integrations > Push Notifications**.
2. Click on the **slider** to enable your webhook.

Your webhook servlet is now enabled.

## Trigger Webhook Callbacks

Your webhook servlet should be ready to receive webhook calls. Try creating @mentions at the Alias User you've created. The webhook server should be triggered and automatically post a reply comment to the @mention on Jam.

You may also go to the **Push Notification Subscription** page to see the status of the webhook server. If the webhook server is responding to Jam incorrectly, an error message will be posted.

## Push Notifications

Push notification subscriptions are a mechanism for capturing events that occur in Jam and sending them to bots or 3rd-party applications.


[+ Push Notification Subscription](#)

☒ **Sample Alias Subscription**
Action ▾

Webhook | Alias Account | # of events 2/6  
Callback URL: <https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer/>

● **Last webhook callback completed successfully**


Push Notification Subscription




**Marc Bell**  
23 minutes ago

**@Webhook Bot** Hello!

[Edit](#) · [Reply](#) · [Like](#) · [Share](#) · [More ▾](#)



**Webhook Bot** I've received a webhook call!  
23 minutes ago · [Like](#) · [Delete](#)



Push Notification Result

SAP BTP also provides a dashboard for the status of your Java servlet. Log into SAP BTP to access the SAP BTP Cockpit. In the list of your Java Applications, you should be able to see your deployed applications and access its server logs. See the [official documentation](#) for more details.

Europe (Trial) / i850769trial / jamwebhooksserver

jamwebhooksserver - Overview

Started

Started at 15 Sep 2016, 13:57:29

Start additional process

Stop

Update

Delete

Application Details

Name: jamwebhooksserver

\*Display Name: jamwebhooksserver

Description:

Edit

Application URLs

<https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer>

Application Maintenance

Not in Maintenance

No Maintenance Application

Start Maintenance

Runtime

Requested during deployment

Java EE 6 Web Profile

2

Availability

N/A

Last HTTP Status Code: N/A

Last Response Time: N/A

Create Check

Most Recent Logging

Type	Process ID	Last Modified	Size	Actions
Default Trace	2bffb4e	15 Sep 2016, 13:58:17	2.18 kB	
HTTP Access Log	2bffb4e	15 Sep 2016, 13:58:11	694 B	
Garbage Collection Log	2bffb4e	15 Sep 2016, 15:09:25	11.0 kB	

Processes

State	Process ID	Compute Unit	Start Time	Actions
	2bffb4e	Lite	15 Sep 2016, 13:57:29	

SAP BTP Java Servlet Dashboard

## 5.1.2 Webhooks - Alias Users - WAR file

Webhooks allow your application to subscribe to certain changes on Jam and receive real time updates.

Whenever a subscribed change has happened, Jam will make a HTTP POST to the callback URL you specify. This eliminates the need for continous requests on our OData API to check for updates, saving you significant network bandwidth and computation costs.

## Try Webhooks now on SAP BTP

We have a sample Java EE servlet packed in a WAR file to demonstrate how webhooks on Jam should be used. This guide will show you how to run the sample servlet on SAP BTP. If you don't have a SAP BTP account, follow [this tutorial](#) to set one up.

## Download the Sample Webhooks Servlet

The following steps will demonstrate how to setup the sample servlet.

1. Download the sample webhooks servlet source code from our [official SAP Github repository](#).
2. Extract the zip file.
3. Navigate to "/Webhooks/Java/SAPJamSampleWebhooksServer".
4. Rename **SAPJamSampleWebhooksServer.war** to **SAPJamSampleWebhooksServer.zip**.
5. Extract the zip file.
6. In the WEB-INF folder, open **web.xml** with a text editor.

## Deploy Your Servlet on SAP BTP to get the Application URL

You will need to deploy your servlet on SAP BTP to get the Application URL:

1. Within SAP BTP, select **Applications > Java Applications**.
2. Click **Deploy Application**.
3. Select **SAPJamSampleWebhooksServer.war**.
4. Click **Deploy**.
5. Click **Start**.
6. Copy and paste the Application URL into a text editor and add a "/" to the end of it - "Application URL/".

Your server should be now deployed with your configuration changes.

## Create an Alias User on Jam

We want to trigger a callback whenever an Alias user is @mentioned. Let's create an Alias user first:

1. In SAP Jam, click on the "cog" settings icon and select **Admin**. The SAP Jam Admin page will appear.
2. Select **Users > Alias Accounts**.
3. Click **New Alias Account**.
4. Create an Alias Account Name in the **Alias Account Name** field.
5. Click **Add an OAuth2 Access Token**. If you do not have an OAuth2 Access Token [click here \[page 14\]](#) to learn how.
6. Select an **OAuth Client** from the drop down menu and click **OK**.
7. Click **Save changes**.

8. Copy the new **OAuth2 Access Token** and paste it as the JAM\_OAUTH\_TOKEN param-value in **web.xml**.
9. Save **web.xml**.

**Manage Alias Account**

Basic Profile Information

Alias UserID: iG51Kp6JiLHXA9P30frdcc

Alias Account Name: Webhook Bot

Description:

Alias Account Name

API Access

OAuth2 Access Token: [Masked]

For OAuth Client: Webhooks

Delete OAuth2 Access Token

OAuth2 Access Token

## Create a new Push Notification Subscription on Jam

With an alias user created, we could create a new push notification subscription that triggers webhook calls whenever the alias user is @mentioned.

1. In the admin settings page select **Integrations > Push Notifications**.
2. Click **+Push Notification Subscription**.
3. Create a name for the Push Notification in the **Name** field.
4. Paste the Application URL from your text editor into the **Callback URL** field.
5. Set the **Scope** to **Alias Account** and enter the alias user we just created in the **Alias Account** field.
6. Enable all the **Alias Account Notifications** in **Event List**.
7. Copy the **Token** field and paste it as the JAM\_WEBHOOK\_VERIFICATION\_TOKEN param-value in **web.xml**.
8. Save **web.xml**.
9. Click **Save**.

**Manage Push Notification Subscription**

Subscription Type: Webhook ☐ Enabled

Name\*: Sample Alias Subscription

Description:

Name



Callback, Token, Scope, Event List

## Redeploy Your Servlet on SAP BTP

You should now have the OAuth and verification token properly configured in web.xml. Let's redeploy your servlet on SAP BTP:

1. Zip the contents back up to **SAPJamSampleWebhooksServer.zip**.
2. Rename this back to **SAPJamSampleWebhooksServer.war**.
3. Within SAP BTP, select **Applications > Java Applications**.
4. Select **SAPJamSampleWebhooksServer**.
5. Click **Update**.
6. Click **Browse...**.
7. Select **SAPJamSampleWebhooksServer.war**.
8. Click **Open**.
9. Click **Update**.
10. Click **Start**.

Your server should be now deployed with your configuration changes.

## Enable Your Webhook in Jam

Let's enable your webhook in Jam:

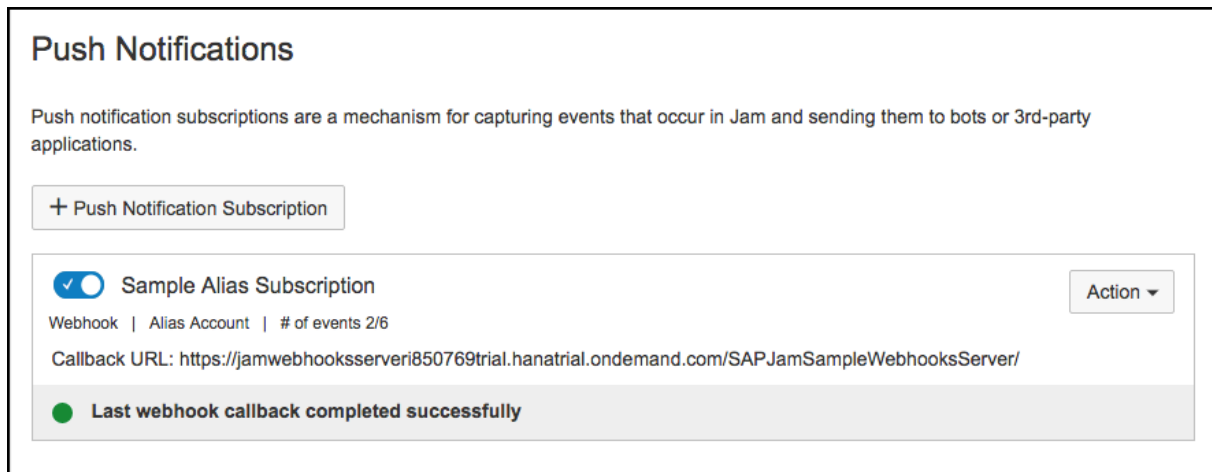
1. Now go back to Jam and select **Integrations > Push Notifications**.
2. Click on the **slider** to enable your webhook.

Your webhook servlet is now enabled.

## Trigger Webhook Callbacks

Your webhook servlet should be ready to receive webhook calls. Try creating @mentions at the Alias User you've created. The webhook server should be triggered and automatically post a reply comment to the @mention on Jam.

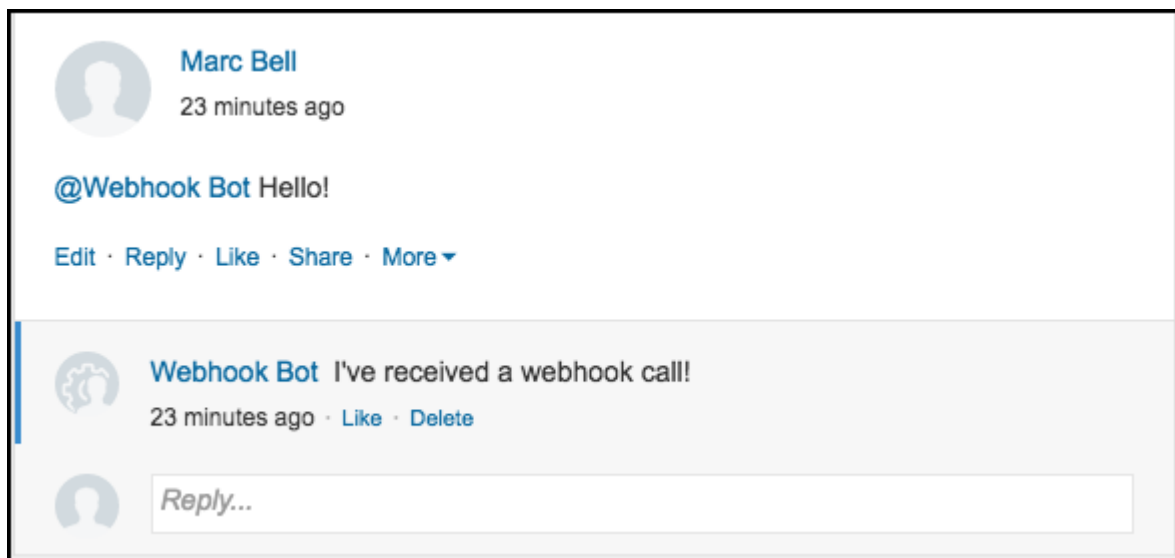
You may also go to the **Push Notification Subscription** page to see the status of the webhook server. If the webhook server is responding to Jam incorrectly, an error message will be posted.



The screenshot shows the 'Push Notifications' section of a dashboard. It includes a description: 'Push notification subscriptions are a mechanism for capturing events that occur in Jam and sending them to bots or 3rd-party applications.' Below this is a button '+ Push Notification Subscription'. A table lists a subscription named 'Sample Alias Subscription' with a toggle switch, an 'Action' dropdown, and details for the webhook URL and event count. A status bar at the bottom indicates 'Last webhook callback completed successfully' with a green dot.

Subscription Name	Status	Action	Webhook	Alias Account	# of events
Sample Alias Subscription	<input checked="" type="checkbox"/>	Action ▼	Webhook	Alias Account	2/6
Callback URL: <a href="https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer/">https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer/</a>					
Last webhook callback completed successfully					

Push Notification Subscription



The screenshot shows a Jam post by 'Marc Bell' from 23 minutes ago. The post content is '@Webhook Bot Hello!'. Below the post are interaction options: 'Edit · Reply · Like · Share · More ▼'. A comment from 'Webhook Bot' is shown below the post, stating 'I've received a webhook call!' with 'Like' and 'Delete' options. At the bottom, there is a 'Reply...' input field.

Push Notification Result

SAP BTP also provides a dashboard for the status of your Java servlet. Log into SAP BTP to access the SAP BTP Cockpit. In the list of your Java Applications, you should be able to see your deployed applications and access its server logs. See the [official documentation](#) for more details.

Europe (Trial)

i850769trial

jamwebhooksserver

jamwebhooksserver - Overview

Started

Started at 15 Sep 2016, 13:57:29

Start additional process

Stop

Update

Delete

Application Details

Name: jamwebhooksserver

\*Display Name: jamwebhooksserver

Description:

Edit

Application URLs

<https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer>

Application Maintenance

Not in Maintenance

No Maintenance Application

Start Maintenance

Runtime

Requested during deployment

Java EE 6 Web Profile

2

Availability

N/A

Last HTTP Status Code: N/A

Last Response Time: N/A

Create Check

Most Recent Logging

Type	Process ID	Last Modified	Size	Actions
Default Trace	2bffb4e	15 Sep 2016, 13:58:17	2.18 kB	
HTTP Access Log	2bffb4e	15 Sep 2016, 13:58:11	694 B	
Garbage Collection Log	2bffb4e	15 Sep 2016, 15:09:25	11.0 kB	

Processes

State	Process ID	Compute Unit	Start Time	Actions
	2bffb4e	Lite	15 Sep 2016, 13:57:29	

SAP BTP Java Servlet Dashboard

## 5.1.3 Webhooks - Groups

Webhooks allow your application to subscribe to certain changes on Jam and receive real time updates.

Whenever a subscribed change has happened, Jam will make a HTTP POST to the callback URL you specify. This eliminates the need for continous requests on our OData API to check for updates, saving you significant network bandwidth and computation costs.

## Get Started with Webhooks

We have a sample Java EE servlet to demonstrate how webhooks on Jam should be used. This guide will show you how to run the sample code on SAP BTP. If you don't have a SAP BTP account, follow [this tutorial](#) to set one up.

You will also need Eclipse (Java EE Edition) with SAP BTP Tools Plugin installed. Follow [this guide](#) if you don't have that set up.

## Download the Sample Webhooks Servlet

The following steps will demonstrate how to setup the sample webhooks servlet.

1. Download the sample webhooks servlet source code from our [official SAP Github repository](#).
2. Extract the zip file.

## Import the JamSampleWebhookClient Project

1. Open Eclipse.
2. Right click in the **Project Explorer** panel and select **Import > Import...**
3. Select **Maven > Existing Maven Projects**.
4. Click **Next**.
5. Click **Browse...**
6. Navigate to `/Webhooks/Java/JamWebHooksTester4j` and click **Open**.
7. Click **Finish**.

## Deploy the Servlet

Although the servlet is not ready yet, we want to deploy it first so we can get a public URL to set up a webhook push notification subscription on Jam.

1. In the **Project Explorer** expand the tree as follows: **SAPJamSampleWebhooksServer > Java Resources > src > com.sap.jam.webhooks.sample**
2. Right click on **WebhookServlet.java** and select **Run As > Run on Server**.
3. Select **Manually define a new server**.
4. Choose **SAP > SAP BTP** as the server type.
5. Enter **hanatrial.ondemand.com** as your Landscape host (or **hana.ondemand.com** if you are using a non-trial account).
6. Click **Next**.
7. Enter your application name in the **Application name** field.
8. Enter your SAP BTP account information in the **Account information** fields.

9. Click **Finish**.

Once deployment is done, your browser should automatically open a link to the newly deployed server with a basic message. Take note of the URL.

## Create a Wiki Page in a Jam Group

We want to create a Wiki page in a new Jam group:

1. Create a new Jam group. (See the [SAP Jam Group Administration Guide](#).)
2. [Create a Wiki page](#) in your new Jam group.

## Create an OAuth2 Token

We want to create an OAuth2 token to provide the servlet access to Jam:

1. [Create an OAuth2 Token \[page 14\]](#)
2. Copy the new **OAuth2 Access Token** and paste it as the JAM\_OAUTH\_TOKEN static string in **WebhookServlet.java**.

## Create a new Push Notification Subscription on Jam

With an alias user created, we could create a new push notification subscription that triggers webhook calls whenever the alias user is @mentioned.

1. In the admin settings page select **Integrations > Push Notifications**.
2. Click **+Push Notification Subscription**.
3. Create a name for the Push Notification in the **Name** field.
4. Paste the URL of the Java Servlet you just deployed on SAP BTP (https://Servlet\_URL/) into the **Callback URL** field.
5. Set the **Scope** to **Group** and enter the name of the group we just created in the **Group** field.
6. Enable **Wiki edit** in **Content Notification** in the **Event List**.
7. Copy the **Token** field and paste it as the JAM\_WEBHOOK\_VERIFICATION\_TOKEN static string in **WebhookServlet.java**.
8. Click **Save**.

### Manage Push Notification Subscription

Subscription Type Webhook ▼ ☐ Enabled

Name\*

Description

Name

Callback URL\*

Token\*  Re-generate

Scope\* Group Group Push Notification Sample ✕ ▼

☐ Alias Account

Event List

Alias Account Notification	# of events 0/2 ^
<input type="checkbox"/> Select all <input type="checkbox"/> @mention <input type="checkbox"/> Group invite	
Content Notification	# of events 1/4 ^
<input type="checkbox"/> Select all <input type="checkbox"/> Document creation <input type="checkbox"/> Document edit <input type="checkbox"/> Wiki creation <input checked="" type="checkbox"/> Wiki edit	

Callback, Token, Scope, Event List

## Redeploy Your Servlet on SAP BTP

You should now have the OAuth and verification token properly configured in the servlet code. Let's redeploy your servlet on SAP BTP:

1. Select the **Servers** tab.
2. Right click on your **SAP BTP server** and select **Publish**.

Your server should be now redeployed with the new changes.

## Enable Your Webhook in Jam

Let's enable your webhook in Jam:

1. Now go back to Jam and select **Integrations > Push Notifications**.
2. Click on the **slider** to enable your webhook.

Your webhook servlet is now enabled.

## Trigger Webhook Callbacks

Your webhook servlet should be ready to receive webhook calls. Edit your wiki you created and publish it. The webhook server should be triggered and automatically post a feed entry in Jam.

You may also go to the **Push Notification Subscription** page to see the status of the webhook server. If the webhook server is responding to Jam incorrectly, an error message will be posted.

## Push Notifications

Push notification subscriptions are a mechanism for capturing events that occur in Jam and sending them to bots or 3rd-party applications.

+ Push Notification Subscription

✓

Sample Group Subscription

Action ▾

Webhook | Group | # of events 1/6

Callback URL: <https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer/>

●

Last webhook callback completed successfully

### Push Notification Subscription

 **Marc Bell** commented on the wiki [Wiki Edit Test](#)  
less than a minute ago · [Group Push Notification Sample](#)

I've received a webhook call!

[Edit](#) · [Reply](#) · [Like](#) · [Share](#) · [More](#) ▼

### Push Notification Result

SAP BTP also provides a dashboard for the status of your Java servlet. Log into SAP BTP to access the SAP BTP Cockpit. In the list of your Java Applications, you should be able to see your deployed applications and access its server logs. See the [official documentation](#) for more details.

Europe (Trial) / i850769trial / jamwebhooksserver

jamwebhooksserver - Overview

Started Started at 15 Sep 2016, 13:57:29

Start additional process Stop Update Delete

Application Details

Name: jamwebhooksserver

\*Display Name: jamwebhooksserver

Description:

Edit

Application URLs

<https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer>

Application Maintenance

Not in Maintenance No Maintenance Application Start Maintenance

Runtime

Requested during deployment

Java EE 6 Web Profile

2

Availability

N/A

Last HTTP Status Code: N/A

Last Response Time: N/A

Create Check

Most Recent Logging

Type	Process ID	Last Modified	Size	Actions
Default Trace	2bffb4e	15 Sep 2016, 13:58:17	2.18 kB	🔍 ⬇
HTTP Access Log	2bffb4e	15 Sep 2016, 13:58:11	694 B	🔍 ⬇
Garbage Collection Log	2bffb4e	15 Sep 2016, 15:09:25	11.0 kB	🔍 ⬇

Processes

State	Process ID	Compute Unit	Start Time	Actions
✔	2bffb4e	Lite	15 Sep 2016, 13:57:29	🔍 ⏸ ⏹

SAP BTP Java Servlet Dashboard

## 5.1.4 Webhooks - Groups - WAR file

Webhooks allow your application to subscribe to certain changes on Jam and receive real time updates.

Whenever a subscribed change has happened, Jam will make a HTTP POST to the callback URL you specify. This eliminates the need for continous requests on our OData API to check for updates, saving you significant network bandwidth and computation costs.



## Try Webhooks now on SAP BTP

We have a sample Java EE servlet packed in a WAR file to demonstrate how webhooks on Jam should be used. This guide will show you how to run the sample servlet on SAP BTP. If you don't have a SAP BTP account, follow [this tutorial](#) to set one up.

## Download the Sample Webhooks Servlet

The following steps will demonstrate how to setup the sample servlet.

1. Download the sample webhooks servlet source code from our [official SAP Github repository](#).
2. Extract the zip file.
3. Navigate to "/Webhooks/Java/SAPJamSampleWebhooksServer".
4. Rename **SAPJamSampleWebhooksServer.war** to **SAPJamSampleWebhooksServer.zip**.
5. Extract the zip file.
6. In the WEB-INF folder, open **web.xml** with a text editor.

## Deploy Your Servlet on SAP BTP to get the Application URL

You will need to deploy your servlet on SAP BTP to get the Application URL:

1. Within SAP BTP, select **Applications > Java Applications**.
2. Click **Deploy Application**.
3. Select **SAPJamSampleWebhooksServer.war**.
4. Click **Deploy**.
5. Click **Start**.
6. Copy and paste the Application URL into a text editor and add a "/" to the end of it - "Application URL/".

Your server should be now deployed with your configuration changes.

## Create a Wiki Page in a Jam Group

We want to create a Wiki page in a new Jam group:

1. Create a new Jam group. (See the [SAP Jam Group Administration Guide](#).)
2. [Create a Wiki page](#) in your new Jam group.

## Create an OAuth2 Token

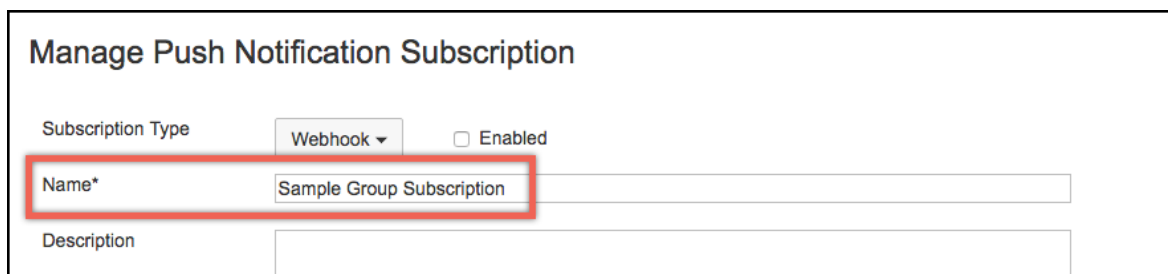
We want to create an OAuth2 token to provide the servlet access to Jam:

1. [Create an OAuth2 Token \[page 14\]](#)
2. Copy the new **OAuth2 Access Token** and paste it as the JAM\_OAUTH\_TOKEN param-value in **web.xml**.

## Create a new Push Notification Subscription on Jam

With an alias user created, we could create a new push notification subscription that triggers webhook calls whenever the alias user is @mentioned.

1. In the admin settings page select **Integrations > Push Notifications**.
2. Click **+Push Notification Subscription**.
3. Create a name for the Push Notification in the **Name** field.
4. Paste the URL of the Java Servlet you just deployed on SAP BTP (`https://Servlet_URL/`) into the **Callback URL** field.
5. Set the **Scope** to **Group** and enter the name of the group we just created in the **Group** field.
6. Enable **Wiki edit** in **Content Notification** in the **Event List**.
7. Copy the **Token** field and paste it as the JAM\_WEBHOOK\_VERIFICATION\_TOKEN param-value in **web.xml**.
8. Save **web.xml**.
9. Click **Save**.



**Manage Push Notification Subscription**

Subscription Type: Webhook ☐ Enabled

Name\*: Sample Group Subscription

Description:

Name

The screenshot shows a configuration form for SAP Jam webhooks. The fields are as follows:

- Callback URL\***: `https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServ`
- Token\***: [Redacted] Re-generate
- Scope\***:
  - ☒ **Group**: `Group Push Notification Sample` ✕
  - ☐ **Alias Account**:
- Event List**:
  - Alias Account Notification** # of events 0/2 ^
    - ☐ Select all
    - ☐ @mention
    - ☐ Group invite
  - Content Notification** # of events 1/4 ^
    - ☐ Select all
    - ☐ Document creation
    - ☐ Document edit
    - ☐ Wiki creation
    - ☒ Wiki edit

Callback, Token, Scope, Event List

## Redeploy Your Servlet on SAP BTP

You should now have the OAuth and verification token properly configured in `web.xml`. Let's redeploy your servlet on SAP BTP:

1. Zip the contents back up to **SAPJamSampleWebhooksServer.zip**.
2. Rename this back to **SAPJamSampleWebhooksServer.war**.
3. Within SAP BTP, select **Applications > Java Applications**.
4. Select **SAPJamSampleWebhooksServer**.
5. Click **Update**.
6. Click **Browse...**.
7. Select **SAPJamSampleWebhooksServer.war**.
8. Click **Open**.
9. Click **Update**.
10. Click **Start**.

Your server should be now deployed with your configuration changes.

## Enable Your Webhook in Jam

Let's enable your webhook in Jam:

1. Now go back to Jam and select **Integrations > Push Notifications**.

- Your webhook servlet is now enabled.

Your webhook servlet should be ready to receive webhook calls. Edit your wiki you created and publish it. The webhook server should be triggered and automatically post a feed entry in Jam.

[illegible]

## Push Notification Subscription

 **Marc Bell** commented on the wiki [Wiki Edit Test](#)  
less than a minute ago · [Group Push Notification Sample](#)

I've received a webhook call!

[Edit](#) · [Reply](#) · [Like](#) · [Share](#) · [More](#) ▼

### Push Notification Result

208 PUBLIC

Europe (Trial)
/
i850769trial
/
jamwebhooksserver

jamwebhooksserver - Overview
☆
🔄
?

Started

Started at 15 Sep 2016, 13:57:29

Start additional process

Stop

Update

Delete

Application Details

Name: jamwebhooksserver

\*Display Name: jamwebhooksserver

Description:

Edit

Application URLs

<https://jamwebhooksserveri850769trial.hanatrial.ondemand.com/SAPJamSampleWebhooksServer>

Application Maintenance

🔧 Not in Maintenance
No Maintenance Application
🔧 Start Maintenance

Runtime

✔ Requested during deployment
Java EE 6 Web Profile
2

Availability

◇ N/A
◇ Last HTTP Status Code: N/A
◇ Last Response Time: N/A
Create Check

Most Recent Logging

Type	Process ID	Last Modified	Size	Actions
Default Trace	2bffb4e	15 Sep 2016, 13:58:17	2.18 kB	📄 ⬇
HTTP Access Log	2bffb4e	15 Sep 2016, 13:58:11	694 B	📄 ⬇
Garbage Collection Log	2bffb4e	15 Sep 2016, 15:09:25	11.0 kB	📄 ⬇

Processes

State	Process ID	Compute Unit	Start Time	Actions
✔	2bffb4e	Lite	15 Sep 2016, 13:57:29	🔄 ⏸ ⏹

SAP BTP Java Servlet Dashboard

## 6 UI5 Embeddable Widgets

### 6.1 UI5 Embeddable knowledge base widget

A highly customizable widget that can be embedded in UI5 compatible SAP applications to provide in-context access to relevant knowledge.

This widget supports:

- Knowledge Base articles as well as other content types such as wikis, documents, etc.
- Customizations such as widget size, columns, sort options, labels, etc.

For example, when embedded in a UI5 service ticket application, the widget can provide the following benefits:

- Recommend relevant Knowledge Base articles depending on the context of the service ticket.
- Provide advanced search with filters for category, tag, group, author, and multiple sort by options.
- Enable users to directly view Knowledge Base articles from within the service ticket experience.
- Supports attaching the URL of a Knowledge Base article to a service ticket (internal link from SAP Jam Collaboration and external link from SAP Jam Communities) and other KB article attributes supported by the defaultColumns property of this widget.

Use the instructions in the following tables to configure your knowledge base widget.

#### Properties

Name	Type	Default Value	Description
jamInstance	object		<p>Parameters for configuring SAP Jam as a knowledge base repository. Both SAP Jam Collaboration and SAP Jam Communities are supported as the jamInstance for searching Knowledge Base Articles or other SAP Jam content. contentSyndicationInstance is the public SAP Jam community that has content syndication enabled for providing public links to the selected Knowledge Base.</p> <p>You must perform the following steps to authenticate into your jamInstance and contentSyndicationInstance using a single use token:</p> <ol style="list-style-type: none"><li>1. <a href="#">Generate an OAuth2 Bearer token. [page 395]</a></li><li>2. <a href="#">Use the OAuth2 Bearer token to generate a single use token. [page 433]</a></li><li>3. Use each single use token as shown in the examples below.</li></ol>

Name	Type	Default Value	Description
<p>Example 1 - SAP Jam Collaboration</p> <pre>jamInstance: {   uri: "https://{jamInstance_domain}",   singleUseToken:     "{jamInstance_singleUseToken}" }</pre> <p>Example 2 - SAP Jam Communities</p> <pre>jamInstance: {   uri: "https://{jamInstance_domain}",   singleUseToken:     "{jamInstance_singleUseToken}",   contentSyndicationInstance: {     uri: "https://{contentSyndicationInstance_domain}",     singleUseToken:       "{contentSyndicationInstance_singleUseToken}",     publicGroup: "{publicGroup_UUID}"   } }</pre> <div> <p><b>Note</b></p> <p>"publicGroup" must be a public group which has been configured as a content syndication target for the Jam collaboration site.</p> </div>			
contentType	string[]	[kb, blog, wiki, document]	<p>Specify types of content in SAP Jam which can be searched in the widget. Available content types:</p> <ul style="list-style-type: none"> <li>• kb</li> <li>• blog</li> <li>• wiki</li> <li>• document</li> </ul>
defaultColumns	string[]	[title, group, author, updatedAt]	<p>Specify what columns of SAP Jam content to be displayed as search result. Available columns:</p> <ul style="list-style-type: none"> <li>• title</li> <li>• group</li> <li>• type</li> <li>• author</li> <li>• createdAt</li> <li>• updatedBy</li> <li>• updatedAt</li> <li>• likesCount</li> <li>• viewsCount</li> </ul>

Name	Type	Default Value	Description
			<ul style="list-style-type: none"> <li>• rating</li> <li>• tags</li> <li>• categories (only for Knowledge Base articles)</li> </ul>
jamGroups	string[]	empty string	If provided, search will be limited to these groups only. Default is empty so that the widget will search all available content on SAP Jam for the user.
defaultSearchMode	string	none	<p>supported values:</p> <ul style="list-style-type: none"> <li>• none (returns nothing if the searchFieldValue property is empty).</li> <li>• all (returns all the results if the searchFieldValue property is empty).</li> </ul>
widgetTitle	string	Suggestions from SAP Jam	Title of the widget.
widgetWidth	sap.ui.core.CSSSize	100%	Width of the widget.
widgetHeight	sap.ui.core.CSSSize	100%	Height of the widget.
previewWidth	sap.ui.core.CSSSize	1000px	Width of content preview panel.
previewHeight	sap.ui.core.CSSSize	700px	Height of content preview panel.
showAdvancedSearch	boolean	false	Set or get the visibility state of the Advanced Search section which contains custom keyword search and filters.
advancedSearchTitle	string	Advanced Search	Text of the Advanced Search section.
searchFieldPlaceholder	string	Search	Text inside custom keyword search box.
filterFields	string[]	[types, groups, authors, categories]	<p>Specify the filters in Advanced Search section. Available filter fields:</p> <ul style="list-style-type: none"> <li>• types</li> <li>• groups</li> <li>• authors</li> <li>• categories</li> <li>• tags</li> </ul>
sortFields	string[]	[relevance, date]	<p>Specify sorting field and method. Available sort fields:</p> <ul style="list-style-type: none"> <li>• relevance</li> <li>• date</li> <li>• rating</li> </ul>
actionColumnTitle	string	Action	Specify the title of column which display action buttons.
pageSize	int	20	Maximum number of search result items on one page.
searchFieldValue	string	empty string	



## Aggregations

Name	Cardinality	Type	Description
buttons	0..n	sap.m.Button	The buttons in the action column.  If this aggregation is not set, the action column will be hide. When press the button, "selectedItem" parameter which will be send back to the event. "selectedItem" will contain the info for content.

## Example for selectedItem

```
{
  "jamContentInfo": {
    "id": "GqBoZJElcNOXWpknW783Pg",
    "type": "Article",
    "title": "SAP Jam Content Sample",
    "content": "Content of SAP Jam Content Sample",
    "group": {
      "group_name": "Collaboration Content",
      "group_id": "x9z4xtZ26gpIqQYICVU5dX"
    },
    "tags": [],
    "categories": [],
    "creator": {
      "creator_name": "Carla Grant",
      "creator_id": "uD0JJ6GaHNbJPoHdRVlgTo"
    },
    "created_at": "2018-12-19T02:59:59Z",
    "updated_at": "2018-12-19T03:00:00Z",
    "is_deleted": false,
    "locale": null,
    "last_updated_by": {
      "updater_name": "Andrew Anderson",
      "updater_id": "uD0JJ6GaHNbJPoHdRVlgTo"
    },
    "href": "http://{jamInstance_domain}/articles/GqBoZJElcNOXWpknW783Pg",
    "views_count": 5,
    "likes_count": 0,
    "rating": 0
  },
  "syndicatedContentInfo": {
    "id": "IY7g2Qj38o5qOfErhYyGBu",
    "href": "http://{contentSyndicationInstance_domain}/articles/IY7g2Qj38o5qOfErhYyGBu",
    "group": {
      "group_name": "Community Group",
      "group_id": "YNvrWwb3igrJ5rEBqEQcHE"
    }
  }
}
```

## How to setup the widget

1. The widget library URL is at: `https://{jamInstance_domain}/widget/jamWidget`
2. Add the library to the project's `resourceRoots`:

```
window["sap-ui-config"] = {  
  ...  
  
  resourceRoots: {  
    "app": "app",  
    "jam.widget": "https://{jamInstance_domain}/widget/jamWidget"  
  },  
  
  ...  
};
```

3. The widget id is "JamKBWidget". Render the widget as a standard UI5 project:

```
<mvc:View ...  
  xmlns:jamWidget="jam.widget" >  
  
  <jamWidget:JamKBWidget id="JamKBWidget" jamInstance="{/jamInstance}"  
widgetWidth="auto"  
  widgetHeight="600px" pageSize="10" defaultColumns="title,group" >  
    <Button text="Attach" press="handleAttach" >  
  </jamWidget:JamKBWidget>  
</mvc:View>
```

# 7 SAP Jam embeddable widgets

## 7.1 About SAP Jam embeddable widgets

SAP Jam Collaboration provides a set of widgets that allow you to embed SAP Jam Collaboration UI screens into HTML pages, thus reducing the complexity of adding SAP Jam functionality into third-party applications. SAP Jam provides two different types of embeddable widgets:

- **SAP Jam div-embedded widgets**—can be hosted in any HTML file by adding a <div> tag with the proper ID and some automatically prepared JavaScript. There are three different div-embedded widgets:
  - The [Feed Widget \[page 215\]](#)
  - The [Recommendations Widget \[page 216\]](#)
  - The [Share Widget \[page 217\]](#)

Samples of each of the above-listed widgets are shown in the section below.

### SAP Jam div-embedded widgets

**SAP Jam div-embedded widgets** are available from a [Widget Builder](#) form in the SAP Jam [Admin](#) console. This mechanism provides easy access to the following widgets, with no need to know JavaScript.

- The **Feed Widget** is a highly customizable feed widget that can be used to embed one of five different types of SAP Jam feeds in an external web page: the Company Feed, My Follows Feed, Group Feed, External Object Feed, or the External Wall Feed.  
**Note:** The minimum width for the feed widget is 320 pixels.

Post Update

Upload File

What's on your mind?

National Rivera North America Account Group

Carla Grant

added the photo [HgsS9OjG6x3hggMtiqBviO.jpg](#)

8 days ago in [National Rivera North America Account Group](#) · [Reply](#) · [Like](#) · [Bookmark](#) · [Delete](#)

Mike Summers

posted the document [National Rivera Information](#)

National Rivera.pptx

Perry Johnson

created the wiki page: [Background](#)

about 1 month ago in [National Rivera North America Account Group](#) · [Reply](#) · [Like](#) · [Bookmark](#) · [Delete](#)

Perry Johnson

created the wiki page: [Documents](#)

about 1 month ago in [National Rivera North America Account Group](#) · [Reply](#) · [Like](#) · [Bookmark](#) · [Delete](#)

Perry Johnson

created the wiki page: [Service Requests](#)

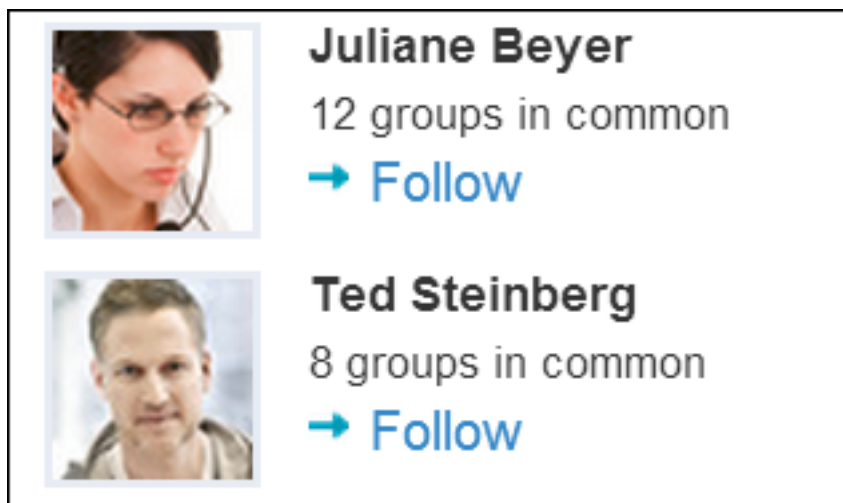
about 1 month ago in [National Rivera North America Account Group](#) · [Reply](#) · [Like](#) · [Bookmark](#) · [Delete](#)

#### Example Feed Widget

- The **Recommendations Widget** is a customizable recommendation widget that can be used to embed SAP Jam content, people, or groups recommendations in an external web page.

216 PUBLIC

SAP Jam Collaboration Developer Guide  
SAP Jam embeddable widgets



Example Recommendations Widget

- The **Share Widget** can be used to embed a "share link" (like other social network share buttons), and which will post the containing page to either SAP Jam group or member feeds when clicked in the external web page.



Example Share Widget

## 7.1.1 SAP Jam Collaboration div-embedded widgets

HTML div-embedded widgets are available to company administrators from the SAP Jam Collaboration [Admin](#) console. The widget builders currently available are the [Feed Widget Builder](#), the [Recommendations Widget Builder](#), and the [Share Widget Builder](#). The options for these widgets can be set using the form for each widget, and the options selected will alter the auto-generated JavaScript that is shown at the bottom of the page. This JavaScript can then be copied and pasted into an HTML page to show the selected SAP Jam content in that page.

The general procedure for accessing the Widget Builders, setting the options that you want, and pasting the resulting JavaScript into your HTML page are described in the following generalized procedure. This page provides a form from which you can set the configuration you want for the widget. Once you have set the widget options that you want, click on [Preview](#) to see both a rendering of options you have selected and the JavaScript for the widget with those options. Copy all of this JavaScript into the body of your HTML page, and add a div tag above the script, with the ID that you set in the [Widget div ID](#) field of the form and with a specified width and height.

**Note:** The minimum width for the feed widget is 320 pixels.

1. In SAP Jam, as a company administrator, click on the cog icon at the top of the page and select [Admin](#) from the context menu.

The SAP Jam [Admin](#) console is displayed.

- From the [Admin](#) console sidebar menu, select [Widget Builders](#).

The [Admin](#) [Widget Builders](#) page is displayed.

## Widget Builders

Feed Widget Builder

Recommendations Widget Builder

Share Widget Builder

Widget div ID

myDiv

ID of the div element that will contain the feed widget

Authentication type

Pre-existing JAM session

The widget will use an existing JAM session. It will show an error message if a session has not already been established.

Feed type

My Follows Feed

Style

JAM

☒ Show Profile Photos in Feed

☐ Show User Profile Photo

☐ Live Feed Updates

☐ Mobile Mode

☐ Filter by the Following Hash Tags

If this option is enabled, the widget will automatically update the feed instead of displaying a "See N New Items" message.

This mode disables features that may not be supported on mobile devices, such as file uploads.

Post Mode

Inline

Reply Mode

Inline

☐ Hide Like Links

☐ Hide Bookmark Links

☐ Limit feed items to a maximum of:

5

Preview

Preview

Code

```
<script type="text/javascript" src="https://integration2.sapjam.com/assets/feed_widget_v1.js"></script>
<script type="text/javascript">
sapjam.feedWidget.init("https://integration2.sapjam.com/widget/v1/feed",
"session");
var w = sapjam.feedWidget.create("myDiv", {type: "follows", avatar: false,
post_mode: "inline", reply_mode: "inline"});
</script>
```

### Widget Builders form

- Click on the tab for the type of widget that you want to generate, either the [Feed Widget Builder](#), the [Recommendations Widget Builder](#), or the [Share Widget Builder](#).
- Select the options that you want for your particular widget type.  
Information about the options that you can select for each widget type are provided in the following widget-specific pages:
  - [Use the Feed Widget Builder \[page 220\]](#)
  - [Use the Recommendations Widget Builder \[page 225\]](#)

- [Use the Share Widget Builder \[page 228\]](#)
  - 5. Once you have selected the options that you want for your particular widget, click [Preview](#), and copy the JavaScript from the [Code](#) text box.
  - 6. Paste the copied JavaScript into your HTML file according to the instructions in the [Add div-embedded widgets to HTML pages \[page 230\]](#) page.
- These options are discussed at length in the next page.

If you have completed the preceding steps, proceed to the [Widget authentication \[page 219\]](#) page.

## 7.1.1.1 Widget authentication

In the second field of the [Feed Widget Builder](#) or the [Recommendations Widget Builder](#), you must select the [Authentication type](#). Authentication ensures that only the people who are entitled to view your organization's SAP Jam Collaboration information can view it.

Of the supported authentication types only one is recommended and three are retained for legacy support.

### Recommended authentication type

- **Single Use Token:** The widget uses a temporary authentication token provided by the SAP Jam Authorization API. See [OAuth 1.0a Authorization API Reference \[page 418\]](#), and particularly the [\[POST\] /oauth/request\\_token \[page 419\]](#).  
A single-use token can be used to obtain a valid Jam Collaboration UI session. Once the token has been used, it is no longer valid. A single-use token can be generated by performing a [\[POST\] /v1/single\\_use\\_tokens \[page 433\]](#) API call. For example, `https://jam4.sapjam.com/v1/single_use_tokens`. This endpoint uses OAuth authentication.  
For more information on how to generate an OAuth token and retrieve a single use token please see [Authentication and Authorization API \[page 395\]](#).  
After obtaining a single-use token, it needs to be included in the parameters passed into the `sapjam.feedWidget.create` function call in the widget initialization script. For example:

```
sapjam.feedWidget.init(
  "https://jam4.sapjam.com/widget/v1/feed", "single_use_token");
var w = sapjam.feedWidget.create("myDiv",
  {type:
    "follows",
    avatar: false,
    post_mode: "inline",
    reply_mode: "inline",
    single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
  }
);
```

## Legacy Authentication Types

### Note

The following authentication types are not recommended for use in production.

- **Log in from your Identity Provider:** The widget uses an existing SAP Jam session if one is available. If the session is not yet established, your page is replaced by your Identity Provider's Sign In page. After signing in, the user is returned to your page.
- **Log In from your Identity Provider in pop-up:** The widget uses an existing SAP Jam session if one is available. If the session is not yet established, a new window or browser tab shows your Identity Provider's Sign In page. After signing in, the window closes and the widget is displayed.
- **Pre-existing SAP Jam session:** The widget uses an existing SAP Jam session. It shows an error message if a session has not already been established.

When you have selected the [Authentication type](#) that you will use, continue to configure your div-embedded widget according to the instructions in the [Use the Feed Widget Builder \[page 220\]](#) or the [Use the Recommendations Widget Builder \[page 225\]](#) page.

### 7.1.1.2 Use the Feed Widget Builder

The Feed Widget Builder provides the easy creation of a highly customizable feed widget that can display many different types of SAP Jam Collaboration feeds in an HTML page, with no requirement for knowing the JavaScript in which they are written.

To create a feed widget with your specifications, do the following:

1. Access the ► [Admin](#) ► [Integrations](#) ► [Widget Builders](#) ► section as described in [SAP Jam Collaboration div-embedded widgets \[page 217\]](#) and click the [Feed Widget Builder](#) tab.  
The [Feed Widget Builder](#) tab content is displayed.



## Widget Builders

---

Feed Widget Builder
Recommendations Widget Builder
Share Widget Builder

---

**Widget div ID**   
ID of the div element that will contain the feed widget

**Authentication type**   
The widget will use an existing JAM session. It will show an error message if a session has not already been established.

**Feed type**

**Style**

☒ Show Profile Photos in Feed

☐ Show User Profile Photo

☐ Live Feed Updates  
If this option is enabled, the widget will automatically update the feed instead of displaying a "See N New Items" message.

☐ Mobile Mode  
This mode disables features that may not be supported on mobile devices, such as file uploads.

☐ Filter by the Following Hash Tags

**Post Mode**

**Reply Mode**

☐ Hide Like Links

☐ Hide Bookmark Links

☐ Limit feed items to a maximum of:

---

**Preview**

**Code**

```
<script type="text/javascript" src="https://integration2.sapjam.com/assets/feed_widget_v1.js"></script>
<script type="text/javascript">
sapjam.feedWidget.init("https://integration2.sapjam.com/widget/v1/feed",
"session");
var w = sapjam.feedWidget.create("myDiv", {type: "follows", avatar: false,
post_mode: "inline", reply_mode: "inline"});
</script>
```

### The Feed Widget Builder

- In the *Widget div ID* field, enter a unique and meaningful name for the HTML div tag that will encapsulate your feed widget.  
This div ID is used in the widget JavaScript that you generate using this form, and it must match the div ID in the HTML page that you use to call and display this widget. Accept the default ID of `myDiv` unless there are other divs in the same page with the same ID, as the div ID must be unique within the page in which you add the widget.
- From the *Authentication Type* drop-down menu, select the type of authentication that you want to use. The options are:
  - Single-use Token*: uses a single-use token provided by the SAP Jam API. **[Recommended]**

For further information generating a single-use token, see the [Auth API](#) section of the *SAP Jam API Documentation*, particularly the documentation of [\[POST\] /v1/single\\_use\\_tokens](#). After obtaining a single-use token, it needs to be included in the parameters passed into the `sapjam.feedWidget.create` function call in the widget initialization script. For example:

```
sapjam.feedWidget.init(
  "https://<jam#>.sapjam.com/widget/v1/feed",
  "single_use_token");
var w = sapjam.feedWidget.create("myDiv", {
  type: "follows",
  avatar: false,
  post_mode: "inline",
  reply_mode: "inline"
  single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
});
```

### Note

The remaining authentication types are not recommended for use in production.

- [Log in from your Identity Provider](#): uses an existing SAP Jam session, or it presents your identity provider's sign-on page, and then redirects to the requested feed widget after the user signs in. **[Not recommended: included for legacy support]**
  - [Log In from your Identity Provider in pop-up](#): uses an existing SAP Jam session, or it presents a SAP Jam sign-on page in a new page or tab, and—upon successful sign-on—the new page closes, and the original requesting page redirects to the requested feed widget. **[Not recommended: included for legacy support]**
  - [Pre-existing Jam Collaboration session](#): uses an existing SAP Jam session, or it displays an error message if one is not found. **[Not recommended: included for legacy support]**
4. From the [Feed type](#) drop-down menu, select the type of feed that you want to display.

The options are:

- [Company Feed](#): will display the same feed that you would see under the Company tab in SAP Jam.
- [My Follows Feed](#): will display the same feed that you would see under your Home tab in SAP Jam.
- [Group Feed](#): will display the same feed, or feeds, for one or more SAP Jam groups. This requires the entry of an additional option, one or more [Group IDs](#):

<div style="display: flex; align-items: center;"> <div style="margin-right: 10px;"><b>Feed type</b></div> <div style="border: 1px solid #ccc; padding: 2px 5px;">Group Feed</div> <div style="margin-left: 5px;">▼</div> </div> <div style="margin-top: 5px;"> <div style="display: flex; align-items: center;"> <div style="margin-right: 10px;">Group IDs:</div> <div style="border: 1px solid #ccc; padding: 2px 5px;">wwA648zAoj0tT39IOHHZdC, RD6jlxCAikvr9raMhUme&lt;</div> </div> </div>
--

Group Feed Widget additional option

If only one group ID is entered in the Group IDs field, the widget displays the same feed displayed on the [Feed Updates](#) tab for that particular group. If multiple group IDs are provided, then the widget displays a drop-down menu from which you can select a specific group or "All groups". Selecting a group displays the feed for that group, and any posts made while viewing that feed are posted to the selected group's wall. If "All groups" is selected, the widget displays the combined feed for all groups. Posting is disabled while viewing the "All groups" feed.

- [External Object Feed](#): will display the feed for an external object or business record. This option requires the entry of two more fields:

<b>Feed type</b>	External Object Feed
External ID:	<input type="text"/>
External type:	<input type="text"/>

#### External Object Feed Widget additional options

- In the [External ID](#) field, enter the value of the [Exid](#) property returned by the OData API's ExternalObjects endpoint. This value is typically the external OData URL for the object.
- In the [External type](#) field, enter the value of the [ObjectType](#) property returned by the OData API's ExternalObjects endpoint. This value is the [External Type](#) value for the object.

The feed will behave differently depending on whether the logged-in user is able to view the full feed history for that particular external object type:

- **Default behavior:** If feed history is not enabled, or the user is not authorized to view the full feed history for the object, they only see the feed updates that are explicitly routed to them, such as via a distribution list.
- **Feed History:** To enable feed history for an external object, a company administrator must configure the necessary authentication settings and external object types in the [External Applications](#) tab of the SAP Jam [Admin](#) page. From this page, administrators can enable the feed history for specific external object types. Once feed history is enabled, any time a user views an external object feed, the Feed Widget calls back to the external application to determine whether the user has permission to view that object. If authorization succeeds, the Feed Widget displays the entire feed history for the external object. If authorization fails, the widget reverts back to the default behavior described above.

For more on External Objects, see [Integrate new business records \[page 7\]](#).

- [External Wall Feed](#): will display the same feed that you would see under the Home tab in SAP Jam if you were to log in as an external user. This option requires the entry of three more fields:

<b>Feed type</b>	External Wall Feed
External ID:	<input type="text"/>
Name:	<input type="text"/>
External URL:	<input type="text"/>

#### External Wall Feed Widget additional options

- In the [External ID](#) field, enter any string that you want to use to uniquely identify the topic.
- In the [Name](#) field, enter a user-friendly name for the topic. This is the name that will show up in the feed when you comment on the topic, which will appear in the form "John Doe commented on <name>".
- [Optional] In the [External URL](#) field, enter any fully-qualified external URL. If this property is provided, then the topic name will be rendered in the feed as a link that opens this URL in a new tab when a user clicks on it.
- [Mentions Feed](#): will display the same feed that you would see in your home page if you click on the "@<your\_user\_name>" tab.
- [Content Item Feed](#): will display the feeds related to specific content items. This option requires the entry of two more fields:

<b>Feed type</b>	Content Item Feed
<b>Content Type:</b>	Document
<b>Content Item ID:</b>	

#### Content Item Feed Widget additional options

- In the *Content Type* field, enter the content type that you want the feed items for. The options are *Document*; *Photo*; *Discussions*, *Ideas*, *Questions*; *Wiki*; or *Blog*.
  - In the *Content Item ID* field, enter the ID of the specific content item for which you want the feed. To get this value, view the page that displays the item, and copy the ID from the last segment of the URL for that page.
- 5. From the *Style* drop-down menu, select the styling that you want to apply to your widget. The options are:
  - Jam
  - SAP Cloud for Customer
- 6. Set the following checkbox toggles as desired:
  - *Show Profile Photos in Feed*: shows thumbnails of the creators of each post in the feed.
  - *Show User Profile Photo*: shows a thumbnail of the requesting user at that top of the feed.
  - *Live Feed Updates*: sets whether updates are automatic or by notification message.
  - *Mobile Mode*: presents only mobile-capable features in the feed.
  - *Filter by the Following Hash Tags*: allows you to set some filtering on the feed based on the hashtags entered in the text box below this toggle.
- 7. From the *Post Mode* drop-down menu, select the option that you want for users' posts. The "Post Mode" refers to the text entry box at the top of the feed in which you can post a comment. The options for this are:
  - *Inline*: A text entry box is displayed at the top of the Feed Widget.
  - *Link to Jam*: A link to the SAP Jam feed is displayed at the top of the Feed Widget.
  - *Hidden*: There is no option for posting displayed at the top of the Feed Widget.
- 8. From the *Reply Mode* drop-down menu, select the option that you want for users' replies. The "Reply Mode" refers to the options to reply to other peoples' comments. The options are:
  - *Inline*: A control, labeled "Reply" is displayed below a user's comment, which opens a Reply text box if clicked.
  - *Link to Jam*: A link to the SAP Jam feed is displayed below a user's comment.
  - *Hidden*: There is no option for replying displayed below a user's comment.
- 9. Set the following checkbox toggles as desired:
  - *Hide Like Links*: this toggle shows or hides users' likes as a post in the feed as replies.
  - *Hide Bookmark Links*: this toggle shows or hides users' bookmarks in the feed as replies.
  - *Limit feed items to a maximum of*: allows you to set the maximum number of feed items.
- 10. At any point in configuring your feed widget, you can click *Preview* to see both a rendered example of the current settings and an updated example of the script that is set by the current options.
- 11. When you are satisfied with the configuration of your widget, copy the JavaScript from the bottom text box of the *Preview* area and paste it into your web page within script tags. Ensure that you click *Preview* immediately before you copy the script to be certain that your most recently selected options are reflected in the script. Also, ensure that you have the div with the ID from step 4 in your HTML page, which calls the script and displays the widget within that div.

The result should look something like the following example:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
feed_widget_v1.js"></script>
    <script type="text/javascript">sapjam.feedWidget.init(
      "https://<jam#>.sapjam.com/widget/v1/feed",
      "single_use_token");
    var w = sapjam.feedWidget.create("myDiv", {
      type: "follows",
      avatar: false,
      post_mode: "inline",
      reply_mode: "inline",
      single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
    });</script>
  </body>
</html>
```

12. Add a div tag above the JavaScript that you pasted into the HTML body in the previous step. This div tag must:

- Be placed before the JavaScript, or you must use something like the jQuery `.ready()` function, so that the order of script elements in the page does not matter.
- Use the same "id" as you set or accepted in the Widget div ID field of the Widgets Builder form.
- Include style statements that set the width and height of the widget, otherwise it will inherit the dimensions of its container element.
  - **Note:** The minimum width for the feed widget is 320 pixels.

The result should look something like this:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <div id="myDiv" style="width:100%; height:100%"></div>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
feed_widget_v1.js"></script>
    <script type="text/javascript">sapjam.feedWidget.init(
      "https://<jam#>.sapjam.com/widget/v1/feed",
      "single_use_token");
    var w = sapjam.feedWidget.create("myDiv", {
      type: "follows",
      avatar: false,
      post_mode: "inline",
      reply_mode: "inline",
      single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
    });</script>
  </body>
</html>
```

### 7.1.1.3 Use the Recommendations Widget Builder

The Recommendations Widget Builder provides the easy creation of a customizable recommendations widgets for SAP Jam Collaboration content, people, or groups recommendations that can be placed into any HTML page, with no requirement for knowing the JavaScript in which they are written.

To create a recommendations widget with your specifications, do the following:

1. Access the **Admin > Integrations > Widget Builders** section as described in [SAP Jam Collaboration div-embedded widgets \[page 217\]](#) and click the *Recommendations Widget Builder* tab. The *Recommendations Widget Builder* tab content is displayed.

**Widget Builders**

Feed Widget Builder   Recommendations Widget Builder   Share Widget Builder

**Widget div ID**   
ID of the div element that will contain the feed widget

**Authentication type**   
The widget will use an existing JAM session. It will show an error message if a session has not already been established.

**Type**

**# of Recommendations**

**Recommendation Type** ☒ Content ☐ People ☐ Groups

[Preview](#)

**Preview**

**Code**

```
<script type="text/javascript" src="https://integration2.sapjam.com/assets/recommendation_widget_v1.js"></script>
<script type="text/javascript">
sapjam.recommendationsWidget.init("https://integration2.sapjam.com/widget/v1/recommendation", "session");
var w = sapjam.recommendationsWidget.create("myDiv", {type: "content", max: "3"});
</script>
```

#### The Recommendations Widget Builder

2. In the *Widget div ID* field, enter a unique and meaningful name for the HTML div tag that will encapsulate your recommendations widget.  
This div ID is used in the widget JavaScript that you generate using this form, and it must match the div ID in the HTML page that you use to call and display this widget. Accept the default ID of `myDiv` unless there are other divs in the same page with the same ID, as the div ID must be unique within the page in which you add the widget.
3. From the *Authentication Type* drop-down menu, select the type of authentication that you want to use. The options are:
  - *Single-use Token*: uses a single-use token provided by the SAP Jam API. **[Recommended]**  
For further information generating a single-use token, see the [Auth API](#) section of the *SAP Jam API Documentation*, particularly the documentation of [\[POST\] /v1/single\\_use\\_tokens](#).  
After obtaining a single-use token, it needs to be included in the parameters passed into the `sapjam.feedWidget.create` function call in the widget initialization script. For example:

```
sapjam.feedWidget.init(
  "https://<jam#>.sapjam.com/widget/v1/feed",
  "single_use_token");
var w = sapjam.feedWidget.create("myDiv", {
  type: "follows",
  avatar: false,
  post_mode: "inline",
  reply_mode: "inline"
  single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
```

```
});
```

### Note

The remaining authentication types are not recommended for use in production.

- [Log in from your Identity Provider](#): uses an existing SAP Jam session, or it presents a SAP Jam sign-on page, and then redirects to the requested feed widget after the user signs in. **[Not recommended: included for legacy support]**
  - [Log In from your Identity Provider in pop-up](#): uses an existing SAP Jam session, or it presents a SAP Jam sign-on page in a new page or tab, and—upon successful sign-on—the new page closes, and the original requesting page redirects to the requested feed widget. **[Not recommended: included for legacy support]**
  - [Pre-existing Jam Collaboration session](#): uses an existing SAP Jam session, or it displays an error message if one is not found. **[Not recommended: included for legacy support]**
4. From the [Type](#) drop-down menu, select the layout of the recommendations that you want to use.  
The options are:
    - [Grid](#)
    - [List](#)
  5. From the [# of Recommendations](#) drop-down menu, select the number of recommendations that you want displayed.  
The options are:
    - 3
    - 5
    - 10
  6. From the [Recommendation Type](#) options, select the radio button for the type of recommendations that you want to display.  
The options are:
    - [Content](#)
    - [People](#)
    - [Groups](#)
  7. At any point in configuring your recommendations widget, you can click [Preview](#) to see both a rendered example of the current settings and an example of the script that is set by the current options.
  8. When you are satisfied with the configuration of your widget, copy the JavaScript from the bottom text box of the [Preview](#) area and paste it into your web page within script tags. Ensure that you click [Preview](#) immediately before you copy the script to be certain that your most recently selected options are reflected in the script. Also, ensure that you have the div with the ID from step 4 in your HTML page, which calls the script and displays the widget within that div.  
The result should look something like the following example:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
recommendation_widget_v1.js">
    </script>
    <script type="text/javascript">sapjam.recommendationsWidget.init(
      "https://<jam#>.sapjam.com/widget/v1/recommendation",
      "single_use_token");
    var w = sapjam.recommendationsWidget.create("myDiv", {
      style: "link",
```

```

        type: "people",
        max: "5",
        single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
    });</script>
</body>
</html>

```

9. Add a div tag above the JavaScript that you pasted into the HTML body in the previous step. This div tag must:

- Be placed before the JavaScript, or you must use something like the jQuery `.ready()` function, so that the order of script elements in the page does not matter.
- Use the same "id" as you set or accepted in the Widget div ID field of the Widgets Builder form.
- Include style statements that set the width and height of the widget, otherwise it will inherit the dimensions of its container element.

The result should look something like this:

```

<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <div id="myDiv" style="width:100%; height:100%"></div>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
recommendation_widget_v1.js">
    </script>
    <script type="text/javascript">sapjam.recommendationsWidget.init(
      "https://<jam#>.sapjam.com/widget/v1/recommendation",
      "single_use_token");
    var w = sapjam.recommendationsWidget.create("myDiv", {
      style: "link",
      type: "people",
      max: "5",
      single_use_token: '883a5486-62af-407a-a5ca-cec6dcf259dc'
    });</script>
  </body>
</html>

```

## 7.1.1.4 Use the Share Widget Builder

The Share Widget Builder creates a share widget that can be embedded in any external web page to add a "share link" (like other social network share buttons) that will post an entry about the containing page to either SAP Jam Collaboration member or group feeds. Creating the share widget has no requirement for knowing how to write JavaScript code.

To create a share widget with your specifications, do the following:

1. Access the **Admin** > **Widget Builders** section as described in [SAP Jam Collaboration div-embedded widgets \[page 217\]](#) and click the *Share Widget Builder* tab. The *Share Widget Builder* tab content is displayed.



## Widget Builders

---

Feed Widget Builder

Recommendations Widget Builder

Share Widget Builder

---

**Widget element ID**

The ID of the HTML DOM element that will open the widget dialog when clicked. Alternatively, a default widget button can be created at this element's location.

☒ **Create default widget button**  
Create a default button that will be appended to the specified DOM element.

**Page URL**

**Group ID (optional)**

☐ **Use custom branding**  
Show your company logo in the widget.

[Preview](#)

---

**Preview**

**Code**

```
<script type="text/javascript" src="https://integration2.sapjam.com/assets/share_widget_v1.js"></script>
<script type="text/javascript">
var w = sap.jam.shareWidget.create("https://integration2.sapjam.com/c/sfcubetree01.com/widget/v1/share", "myDiv", {create_button: true});</script>
```

### The Share Widget Builder

- In the *Widget element ID* field, enter a unique and meaningful name for the HTML div tag that will encapsulate your Share widget.  
This div ID is used in the widget JavaScript that you generate using this form, and it must match the div ID in the HTML page that you use as the container for this widget. Accept the default ID of `myDiv` unless there are other divs in the same page with the same ID, as the div ID must be unique within the page in which you add the widget.
- Select the *Create default widget button* to have a widget button included in the embedded widget.

#### i Note

If you deselect this option, you must add the text or icon to be displayed in the div tag from which the widget is displayed.

- In the *Page URL* text box, enter the URL of the external web page in which the widget is embedded. It is the initial content of this page that will be displayed in the SAP Jam feed.
- Optionally, in the *Group ID (optional)* text box, enter the ID for the group in which you want the external page to have an entry added to that group's feed (group wall).  
You can get this ID from the last segment of the group's URL. For example, in the group URL, `https://<jam#>.sapjam.com/groups/about_page/6jAz17p5W9gtIyEv6O7TAQ`, you would use `"6jAz17p5W9gtIyEv6O7TAQ"`.

#### i Note

If you leave this field blank, the widget will add the shares to your company's members' walls.

- To include your company logo in the shared content, select the *Use custom branding* option. This is the logo configured in your SAP Jam's [Admin > Branding and Support options](#).

7. At any point in configuring your share widget, you can click [Preview](#) to see both a rendered example of the current settings and an example of the script that is set by the current options.
8. When you are satisfied with the configuration of your widget, copy the JavaScript from the bottom text box of the [Preview](#) area and paste it into your web page. Your script should look something like this:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
share_widget_v1.js"></script>
    <script type="text/javascript">var w = sap.jam.shareWidget.create(
      "https://<jam#>.sapjam.com/c/sfcubetree01.com/widget/v1/share",
      "myDiv", {
        url: "http://www.example.com/some-page.html",
        create_button: true,
        branding: true
      });</script>
  </body>
</html>
```

Also, ensure that you have the div with the ID from step 3 in your HTML page, and that you call the script from within that div. For example, you could paste the following into the body of an external web page:

```
<html>
  <head>
    <title>Some Page</title>
  </head>
  <body>
    <div id="myDiv" style="width:100%; height:100%"></div>
    <script type="text/javascript" src="https://<jam#>.sapjam.com/assets/
share_widget_v1.js" ></script>
    <script type="text/javascript">var w = sap.jam.shareWidget.create(
      "https://<jam#>.sapjam.com/c/sfcubetree01.com/widget/v1/share",
      "myDiv",
      {
        url: "http://www.example.com/some-page.html",
        create_button: true,
        branding: true
      });</script>
  </body>
</html>
```

### 7.1.1.5 Add div-embedded widgets to HTML pages

This procedure is the conclusion of the previous procedures or the [SAP Jam Collaboration div-embedded widgets \[page 217\]](#) section:

- [Widget authentication \[page 219\]](#)
- One of the following:
  - [Use the Feed Widget Builder \[page 220\]](#)
  - [Use the Recommendations Widget Builder \[page 225\]](#)
  - [Use the Share Widget Builder \[page 228\]](#)

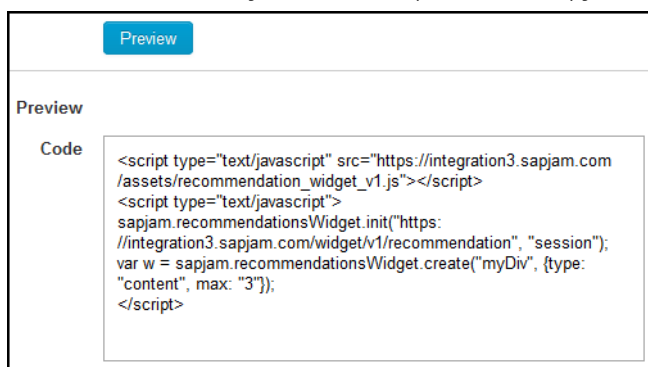
Ensure that you have completed these previous parts of the procedure before continuing.

Embed your div-based widget, generated using the SAP Jam Collaboration [Widget Builder](#) any HTML page by doing the following:

1. Open or create the HTML page in which you want to embed your div-based widget in a text or HTML editor. Only a minimal HTML structure is required, such as this:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
  </body>
</html>
```

2. In the **Admin** > **Widget Builders** tab in which you have finalized the configuration of your Feed or Recommendation widget, click [Preview](#) to generate the JavaScript for your widget in the bottom text box of the [Preview](#) area, with your selected options, and copy the entire contents of that text box.



The Preview area's Code box

3. Paste the JavaScript from the bottom text box of the [Preview](#) area into the body of your web page. The result should look something like this:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <script type="text/javascript"
      src="https://<jam#>.sapjam.com/assets/
recommendation_widget_v1.js">
    </script>
    <script type="text/javascript">
      sapjam.recommendationsWidget.init("https://<jam#>.sapjam.com/
widget/v1/recommendation",
      "session");
      var w = sapjam.recommendationsWidget.create("myDiv", {type:
"content", max: "3"});
    </script>
  </body>
</html>
```

4. Add a div tag above the JavaScript that you pasted into the HTML body in the previous step. This div tag must:
  - Be placed before the JavaScript, or you must use something like the jQuery .ready() function, so that the order of script elements in the page does not matter.
  - Have the same "id" as you set or accepted in the [Widget div ID](#) field of the [Widgets Builder](#) form.
  - Include style statements that set the width and height of the widget, otherwise it will inherit the dimensions of its container element.

The result should look something like this:

```
<html>
  <head>
    <title>SAP Jam Div-Based Widget</title>
  </head>
  <body>
    <div id="myDiv" style="width:100%; height:100%"></div>
    <script type="text/javascript"
      src="https://<jam#>.sapjam.com/assets/
recommendation_widget_v1.js">
    </script>
    <script type="text/javascript">
      sapjam.recommendationsWidget.init("https://<jam#>.sapjam.com/
widget/v1/recommendation",
        "session");
      var w = sapjam.recommendationsWidget.create("myDiv", {type:
"content", max: "3"});
    </script>
  </body>
</html>
```

5. Save your file.

The embedded widget should show the expected SAP Jam content to anyone with the appropriate privileges.

## 8 SAP Jam API

### 8.1 About the SAP Jam API

The SAP Jam Collaboration OData API allows you to integrate SAP Jam Collaboration features into your business critical applications, and it allows you to integrate data from your business critical applications into SAP Jam.

This part of the *SAP Jam Collaboration Developer Guide* provides the following sections of API Documentation:

- Please see the *SAP Jam API Reference* at the following location: <https://jam2.sapjam.com/ODataDocs/ui>
- [SAP Jam OData API Tutorial \[page 234\]](#): This tutorial is designed to give you a good understanding of SAP Jam Collaboration's OData API implementation and to help you to quickly get up to speed using the SAP Jam API.
- [Authentication and Authorization API \[page 395\]](#): Client applications using the SAP Jam REST or OData APIs have two options for providing authorization and authentication for their users: an OAuth1.0a 3-Legged workflow, or SAML assertions from a trusted identity provider. The OAuth1.0a workflow is best for client applications without access to a SAML identity provider (IDP), although it requires some interaction with the end user. The SAML assertions from a trusted IDP configured in the SAP Jam Admin console by your company administrator. Additionally, there is the possibility of single-use tokens, although this approach is better suited to granting immediate, short-term access to single pages.
- [SAP Jam REST API \[page 436\]](#): The SAP Jam Collaboration REST API has mostly been ported to the OData API; however, one set of REST API calls remain: the REST Social Reports API calls. The Social Reports REST API calls allow you to generate and retrieve reports on social activity in SAP Jam.

#### 8.1.1 SAP Jam API Reference

Please see the *SAP Jam API Reference* at the following location: <https://jam2.sapjam.com/ODataDocs/ui>

## 8.1.2 SAP Jam OData API Tutorial

This tutorial is designed to get you up and running with the SAP Jam Collaboration's OData API as quickly as possible. It provides the following concept, example, and in-depth lessons.

Lessons	Using the OData API Reference	Introduction to OData	The Groups and related API	OData Query Parameters	The GroupMembership API	The Member and related API	The Forum and related API	Uploading and downloading files	The ContentItem and related API	The Image and related API	The Comment and related API	Using @mention in the API	The Notification API	The ExternalObject and related API	The GroupGadget and related API	The Task and related API	The Kudos and related API	Understanding OData metadata	Batching API calls together	Supported HTML tags
Type of lesson																				
Concept lessons	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
Example lessons	✗	✗	✓	✓	✓	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗
In-depth lessons	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗

The lessons in this tutorial, and the subjects that they cover, are:

- [Using the OData API Reference \[page 235\]](#): describes the organization of the information for the endpoints, and it shows you how to use the **Try** feature.
- [Introduction to OData \[page 241\]](#): provides a conceptual introduction to OpenData (OData) APIs and instructions on how to use this SAP Jam OData API Tutorial.
- [The Groups and related API \[page 256\]](#): focuses on how to create Jam Groups—directly or from templates—how to retrieve information about them, how to modify them, and how to delete them. It introduces the HTTP headers, URL parameters, and—in some cases—the request payloads required to make normal OData POST, GET, PATCH, and DELETE calls. It also shows the type of returns you can expect from each method.
- [OData Query Parameters \[page 323\]](#): shows you how to manipulate the information that is returned to a collection-valued GET call. Collection-valued calls are GET operations that return the information on multiple resources.
- [Uploading and downloading files \[page 359\]](#): introduces the HTTP headers and URL parameters, including the required query parameters, that are needed to upload and download files.
- [Using @mention in the API \[page 364\]](#): This in-depth lesson describes how at-mentions are handled in the SAP OData API.
- [The Search API \[page 375\]](#): provides a search capability that covers all material within your organization's SAP Jam content.
- [Understanding OData metadata \[page 381\]](#): explains the OData \$metadata file, which is a CSDL file that is made available to clients to help them discover the structure and organization of all the entities (record types), known links between resources (navigations), and the service operations that have been added to manage resources beyond the usual create, retrieve, update, and delete operations.
- [Batching API calls together \[page 386\]](#): explains batch requests, which enable members to bundle OData API requests into one consolidated request and get back a consolidated response. SAP Jam Collaboration supports \$batch requests for GET OData requests. Currently, changesets are not supported as part of the batch request. A maximum of ten requests can be used as part of a single batch request.
- [Supported HTML tags \[page 390\]](#): lists the HTML tags that are supported in various SAP Jam Collaboration OData API inputs such as inputs for wikis, blogs, Questions, Ideas, and Discussions.

## 8.1.2.1 Using the OData API Reference

The SAP Jam OData API Reference provides full documentation for each endpoint, or API call. This page provides a quick description of the information presented in the *SAP Jam OData API Reference*'s sections.

### Learning Objectives

The learning objectives for this page are:

- Be aware of the three [sections \[page 235\]](#) of the *SAP Jam OData API Reference* user interface, and understand what information is displayed in each section, including:
  - Know what information and controls are available in the [banner \[page 236\]](#) section of the API Reference.
  - Know what information and controls are available in the [navigation sidebar \[page 236\]](#) section of the API Reference.
  - Know what information and controls are available in the [content \[page 236\]](#) column of the API Reference.

### Information in the SAP Jam OData API Reference sections

#### i Note

Please see the *SAP Jam API Reference* at <https://jam2.sapjam.com/ODataDocs/ui>.

The SAP Jam OData API Reference is organized into the three sections shown in the following diagram.

SAP Jam OData API Documentation

Token OAuth2 Access Token

Developer Guide API Reference

INTRODUCTION

Overview
OData Concepts
Responses
Metadata

CONTENT

ContentItem
[POST] /ContentItems
[GET] /ContentItems(Id='', ContentItemType='')
[PATCH] /ContentItems(Id='', ContentItemType='')
[DELETE] /ContentItems(Id='', ContentItemType='')
[GET] /ContentItems(Id='', ContentItemType='')/\$value
[PATCH] /ContentItems(Id='', ContentItemType='')/\$value
Navigations
Service Operations
ContentListItem
Folder

ContentItem

A ContentItem can be a wiki Page, a BlogEntry, a Document (an image, sound file, video, or Office document), a Tool (decision, agenda, timeline, or ranking), or a Poll. Blogs differ from wikis in that BlogEntries can only be added by the blog's creator, while a Page can be added to a wiki or modified by anyone with the suitable privileges. Wikis, Blogs, Tools, and Polls are all variations of HTML content, as are Discussions, Ideas, and Questions. ContentItems can be added to a Group in the Content section, to a Folder, or to a Discussion, Idea, or Question as an Attachment.

Request / Response Schema

Type	Property	Description
string	Id GET	Optional Readonly The unique ID of the ContentItem.
string	Name GET PATCH POST	Optional Creatable The name of the ContentItem.
string	ContentItemType GET	Optional Filterable Readonly The type of the ContentItem. There are five supported ContentTypes: Page, BlogEntry, Document, Tool, and Poll.  Can be: BlogEntry Document Page Poll Tool
string	Description GET PATCH POST	Optional Nullable Creatable The description of the ContentItem.
string	CreatedAt GET	Generated Optional Readonly The date and time that the ContentItem was created. This will appear in

## API Reference sections

The sections of the SAP Jam OData API Reference are:

1. The **banner** section contains the title of the API Reference and the **OAuth2 Token** entry field, which allows you to enter the OAuth 2.0 Token generated in the *SAP Jam Collaboration Administrator Guide's OAuth Client* section. See the [SAP Jam API Authorization \[page 237\]](#) page for instructions on how to generate that token.
2. The **navigation sidebar** shows a menu of the Introduction section, all of the entity sections and their endpoints. Click on any entity section title to expand that section and show its information in the content column. Click on any endpoint signature to show its documentation in the content column.
3. The **content column** displays the documentation on each entity grouping (request / response schema, entities, endpoints, navigations and service operations). See the following section in this page, [Information in the Content column \[page 236\]](#) for details on what information this column contains.

## Information in the Content column

The Content column provides the following information:

- Overview - An introduction to the API Reference.
- OData Concepts -Explains concepts that are necessary to effectively use our OData API. These include authorization, paging, expanding and selecting.
- Responses - A table of response codes (200, 201, etc.) with their related methods (GET, POST, PATCH, PUT, DELETE) and descriptions.
- Metadata - A file that is primarily used to provide information to client applications on the API entities, their properties, their navigations, and on the available service operations. This is provided primarily to aid in the



import of data to the client applications. As an XML file, it is human-readable, so you can refer to the `$metadata` file for much of the information about the API. For information on reading the `$metadata` file, please see [Understanding OData metadata \[page 381\]](#).

- Entity groupings - These match the top-level titles shown in the navigation sidebar. These are "entity groupings" in that they frequently cover the API endpoints for more than one entity. For example, the [Group](#) entity set covers endpoints that relate to the **Group**, **GroupTemplates**, **SystemGroupTemplates**, **GroupMemberships**, **GroupExternalObjects**, **GroupGadgets**, and **GroupGadgetObjects** entities. Each of these sections contains:
  - a quick description of the entity grouping
  - the request / response schema for the entity grouping
  - descriptions of each of the entities within the entity grouping
  - navigations for the entity grouping (when applicable)
  - service operations for the entity grouping (when applicable)
- Following the documentation on each entity grouping section is documentation on each endpoint of the entity grouping. That information varies depending on whether it is a POST, GET, PATCH, or DELETE operation.

All endpoint documentation includes:

- The signature of the endpoint, which is the HTTP verb plus the endpoint's resource path; for example: `[GET] /Groups`.
- A short description of the purpose or function of the API call.
- A [Show Options](#) drop-down menu that contains the [Request Headers](#) and [Query Parameters](#) required to make the API call:
  - [Request Headers](#) (required):
    - Authorizaton - Authentication credentials for HTTP authentication.
    - Accept - Desired MIME type of the response (XML, JSON).
  - [Query Parameters](#) (when applicable) - Descriptions and text boxes for inputting the various [Query Parameters](#) (paging, expanding, selecting) that are either required or are available for optional use with the API call.

## 8.1.2.1.1 SAP Jam API Authorization

To use the SAP Jam Collaboration API [Try](#) feature, you must generate and copy an OAuth 2.0 bearer token, and paste that token into the SAP Jam API Reference [OAUTH2 TOKEN](#) text box in the upper-left corner of the API Reference. This page describes how to generate that access token.

### Learning Objectives

The single learning objectives for this page is:

- Understand how to generate an OAuth2 token for use with the *SAP Jam OData API Reference's Try* feature.

## Set up OAuth authentication for the SAP Jam API

1. As an SAP Jam administrator, click on the cog icon (or your name, in a SuccessFactors Platform-integration instance of SAP Jam) at the top of any SAP Jam page and select [Admin](#) from the context menu. The SAP Jam Collaboration Admin console displays.
2. From the left navigation sidebar, select [Integration > OAuth Clients](#). The [OAuth Clients](#) page displays, showing a catalog of the OAuth client instances that have already been configured for your instance of SAP Jam.

OAuth Clients			
OAuth clients registered for your company.			
			<a href="#">Add OAuth Client</a>
Name	Integration URL	Date Added	
Example.com's CRM	<a href="http://sap-crm.example.com">http://sap-crm.example.com</a>	February 27, 2015 11:46 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's ECC (SD)	<a href="https://sap-ecc-sd.example.com">https://sap-ecc-sd.example.com</a>	April 09, 2015 02:27 PM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's C4C	<a href="https://sap-c4c.example.com">https://sap-c4c.example.com</a>	August 10, 2015 11:25 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>

### OAuth Clients catalog

3. If one does not already exist, create an OAuth client configuration for your SAP Jam API use by clicking [Add OAuth Client](#).

#### **i** Note

Any OAuth client configuration should point to the `$metadata` page for the API that will be used. For SAP Jam API use, this should be `https://{jam#}[page 246].sapjam.com/api/v1/OData/$metadata`, where `{jam#}` specifies your organization's SAP Jam data center.

The [Register a new OAuth Client](#) form displays.

### Register a new OAuth Client

Name*	<input type="text" value="SAP Jam API use"/>
Feed Filtering	<input type="text" value="none"/>
Integration URL*	<input type="text" value="https://{jam#}.sapjam.com/api/v1/OData/\$metadata"/> <small>A link to find out more about this application.</small>
Callback URL	<input type="text"/>
Support URL	<input type="text"/>
	<input type="checkbox"/> Can Suppress Notifications
X509 Certificate (Base64)	<input type="text"/> <small>x.509 certificate used to verify the signature of assertions supplied by the OAuth client.</small>

### Administrative Area

Administrative Area	<input type="text" value="Company"/>
---------------------	--------------------------------------

Register a new OAuth Client form

- **[Required]** In the [Name](#) field, enter a meaningful name that will allow other SAP Jam administrators to recognize what this OAuth client is or is used for.
- From the [Feed Filtering](#) drop-down menu, ensure that [none](#) is selected.
- **[Required]** In the [Integration URL](#) field, enter the URL to the client application's API metadata. For the SAP Jam API, this would be `https://{jam#} [page 246].sapjam.com/api/v1/OData/$metadata`, where `{jam#}` specifies your organization's SAP Jam data center.
- Optionally, in the [Callback URL](#) field, enter the callback URL for the client application's API calls. For SAP Jam API use, this field should remain blank.
- Optionally, in the [Support URL](#) field, enter the support URL for the client application's API. For SAP Jam API use, this field should remain blank.
- Optionally, select the [Can Suppress Notifications](#) checkbox to allow the suppression of notifications from external data sources that use this OAuth client. It is up to the developer of this external application integration whether they will disable notifications or not, but this setting determines whether notification suppression will be permitted from this external application. For SAP Jam API use, this check box should remain unselected.
- In the [X509 Certificate \(Base64\)](#) text box, enter the Transport Layer Security (TLS; supercedes SSL) public key certificate string for the client application's API access.

#### i Note

If you provide a X509 certificate, you don't get an OAuth2 token, and you would be signing your requests with this certificate.

For SAP Jam API use, this field should remain blank.

- The [Administrative Area](#) drop-down menu allows you to select the area in which you want this OAuth Client configuration to be available. The default is "Company", which will make it available to all groups and areas. Selecting a specific area will limit the scope of the OAuth Client configuration and will limit the management of that configuration to either area administrators assigned responsibility for that area or to company administrators.

For SAP Jam API use, this drop-down menu should remain at the default of [Company](#).

- When all of the above settings are complete, click [Save](#) to save the record and establish the trust relationship with the OAuth client application.

You are returned to the [OAuth Clients](#) page, with the OAuth client record that you just added listed in the catalog.

4. View the OAuth client records that you just created by clicking [View](#) on the row for your newly SAP Jam API OAuth Client.

An [OAuth Client: {OAuth\\_client\\_name}](#) page is displayed.

## OAuth Client: SAP Jam API use

**Key:** ExampleExampleExamp1

**Secret:** ExampleExampleExampleExampleExampleExamp

**Feed Filtering:** none

**Integration URL:** https://{jam#}.sapjam.com/api/v1/OData/\$metadata

**Callback URL:**

**Support URL:**

**Can Suppress Notifications:** No

**X509 Certificate (Base64):**

We support the PLAINTEXT, HMAC-SHA1 and RSA-SHA1 signature methods.

## OAuth2 Access Token for company admin 'Carla Grant'

Create OAuth2 Access Token

### Administrative Area

This is not in any administrative area (available to the entire company)

[Edit](#) | [Back](#)

View the OAuth Client page

5. Click [Create OAuth2 Access Token](#).

The [OAuth2 Access Token](#) displays above the button, which is now labeled [Delete OAuth2 Access Token](#).

## OAuth2 Access Token for company admin 'Carla Grant'

**OAuth2 Access Token:** AnotherExampleAnotherExampleAnotherExamp

This OAuth2 access token can be used for this company admin for user-based APIs. (for example our [OData APIs](#)).

Delete OAuth2 Access Token

### Generate the OAuth Client page

6. Copy the [OAuth2 Access Token](#) and paste it into the [OAUTH2 TOKEN](#) text box in the upper-left corner of the *SAP Jam OData API* documentation.

### Note

If you, or anyone in your organization with company administrator privileges, deletes the OAuth2 token, it will be invalid for API Reference use, and must be regenerated, copied, and pasted into the [OAuth2 Access Token](#) field in the *SAP Jam OData API Reference* again.

## Learning Objective Review

Have you accomplished the learning objective for this page?

- Do you understand how to generate an OAuth2 token for use with the *SAP Jam OData API Reference*'s **Try** feature?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Introduction to OData \[page 241\]](#).

## 8.1.2.2 Introduction to OData

This section provides an introduction to the Open Data (OData) protocol, which is a highly formalized implementation of the Representational State Transfer (REST) protocol.

























This section contains the following pages of information:

- General information:
  - [Open Data \(OData\) protocol Specifications \[page 242\]](#): provides links to the OASIS OData specification documentation that the SAP Jam OData API is based on.
  - [OData terminology \[page 242\]](#): is drawn from Atom (XML), JSON, and HTTP, although the usage is not always the same.
- [Understanding the OData \\$metadata file \[page 255\]](#): The OData \$metadata file is a CSDL file that is made available to clients to help them discover an OData API's structure and organization.
- [Use of curl commands \[page 255\]](#): This tutorial uses curl commands to summarize the information that must be passed to make an effective API call. A **curl command** (sometimes pronounced "see-URL") provides a concise view of exactly what elements have to be passed in the API call. This is most helpful if you are using `lib-curl` for your encoded API calls, although other options are available.

## 8.1.2.2.1 Open Data (OData) protocol Specifications

The Open Data (OData) protocol was originated by Microsoft and SAP. It is now developed and maintained by OASIS. OData is based on the Atom Publishing protocol. It provides a very formalized, well-defined approach to creating a ReSTful web API. The OData specification requires that APIs be available in either XML (AtomPub) or JSON formats. The SAP Jam OData API conforms to the OData version 2 specification, although it is subject to change in the ways allowed by the OData version 4 specification, with regard to adding new properties, navigations, and entity types. Every effort is made not to break backwards compatibility, but security issues that require a breaking change will be made.

The key specifications used in OData API development and usage are listed here for your reference:

- [RFC 2616, Hypertext Transfer Protocol, HTTP/1.1](#)  
- [RFC 4627, The application/json Media Type for JavaScript Object Notation \(JSON\)](#)  
- [RFC 5023, Atom Publishing Protocol](#)  
- The OASIS [OData Technical Committee](#)  web site 
- [OData.org](#)  
  - [OData Version 2.0](#)  
  - [OData Overview](#)  
  - [OData URI Conventions](#)  
  - [OData Terminology](#)  
  - [OData Operations](#)  
  - [Atom Format](#)  
  - [JSON Format](#)  

## 8.1.2.2.2 OData terminology

OData terminology is a bit confusing as it is drawn from its various constituent technologies, which don't always use the same terms.

To illustrate these issues, the common OData terms, as well as some approximate equivalents from Atom (XML), JSON, and HTTP usage are shown in the following table.

OData (with definition)	Atom (XML)	JSON	HTTP
<b>property:</b> an attribute-value pair that describes a single quality of a resource	property	property	(no equivalent)
<b>entity, resource, or object:</b> a network-accessible data object or service that can be identified by a URI; typically a resource is described by a predefined set of properties (from HTTP)	entry (also used in OData)	object	resource (entity differs)

OData (with definition)	Atom (XML)	JSON	HTTP
<b>EntityType:</b> the abstract data model of a type of resource	(no equivalent)	(no equivalent)	(no equivalent)
<b>collection:</b> a set of resources, often returned to a GET request for a certain type of resource	feed	array of objects	(no equivalent)

Additional terms relevant to OData API usage are:

- **Collection-valued API calls:** are API GET calls to an entity that do not specify a single resource, such as `[GET] /Groups`. These calls will return a feed or collection of resources. For example, the Groups that the currently logged-in user has access to.
- **Binary Large Objects (BLOB):** are binary data that is stored as a single entity (a file), typically graphics, videos, office documents, and PDFs.  
The OData specification describes [Creating Media Link Entries \(MLEs\)](#) in section 2.5 of the Operations page.  
Note that API clients should be able to follow redirect responses, which are required to improve the performance of Content Distribution Networks (CDN).

For more information, see [OData Terminology](#).

### 8.1.2.2.3 HTTP methods

The appropriate HTTP method, or verb, must be set for each API call.

Similarly to most REST implementations, which use POST, GET, PUT, and DELETE HTTP methods to perform create, retrieve, update, and delete (CRUD) operations, the SAP Jam OData API uses POST, GET, PATCH, or DELETE operations.

Both this SAP Jam OData API Tutorial and the SAP Jam OData API Reference show clearly which method must be used for any API call.

Protocol	Create	Retrieve	Update	Delete
ReST	POST	GET	PUT or PATCH	DELETE
SAP Jam OData	POST	GET	PATCH	DELETE

## 8.1.2.2.4 HTTP headers

HTTP headers are used to specify several aspects of an API call's operation.

The HTTP headers used by the SAP Jam OData API include:

- **Authorization: OAuth {OAuth1.Oa\_protocol\_parameters}** This provides user authorization using OAuth 1.0a, for example:

```
Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIHjIm",
oauth_nonce="RPUt7ytQ9w",
oauth_signature_method="HMAC-SHA1", oauth_timestamp="1316135320",
oauth_version="1.0",
oaccess_token="vbXkQoSkiqyYIwxI2I2u", oauth_signature="P5scFJf6CZlBBMELB9kb
%2FvM0ktQ%3D"
```

- **Authorization: Bearer {OAuth\_bearer\_token}** This provides user authorization using OAuth 2.0, for example:

```
Authorization: Bearer As3UvIaYEvdXoeREtmSz3qeCpnNvrrHZhVMswcBV
```

- **Accept: {application/json|application/atom+xml}** This header allows you to specify what is an acceptable response to your API call: the XML or JSON format. This option is preferred to using a query option of `?$format=json`. Note that the default format for OData API calls is XML, so that does not need to be specified unless you want to use JSON. To do so, specify that responses be in JSON format in the HTTP Accept header. For example:

```
Accept: application/json
```

The XML option, while not required, would be:

```
Accept: application/atom+xml
```

- **ContentType: {MIME\_type}** Informs the API that you are sending content in a specific format in the request. This must be specified as the MIME type for the content that you are submitting. Valid options include:
  - `application/json`
  - `application/atom+xml`
  - `text/html`
  - `text/html;type=wiki`
  - `text/html;type=blog`
  - `image/png`
  - `image/jpg`
  - `video/mp4`
  - `application/vnd.ms-powerpoint`
  - `application/vnd.openxml-formats-officedocument.presentationml.presentation`

For example:

```
ContentType: image/png
```

The above examples are all valid, but the list of supported MIME types is far from complete.

- **SAP-JAM-CONTENT-ADMINISTRATION: {true}** This header allows company administrators to enable content administration on their company for this request. With content administration enabled, company



administrators can perform CRUD operations to all groups and content within their company regardless of group membership.

- **Example 1** - Show all Content Items in a group that the company administrator is not a member of.

```
Request Url: GET {{api_url}}/Groups('{{Id}}')/AllContentItems
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

- **Example 2** - Search for all Content Items in a group that the company administrator is not a member of.

```
Request Url: GET {{api_url}}/Search?Query='{{search_item}}'
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

- **Example 3** - Delete an item in a group that the company administrator is not a member of.

```
Request Url: DELETE {{api_url}}/
ContentItems(Id='{{Id}}',ContentItemType='{{Type}}')
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

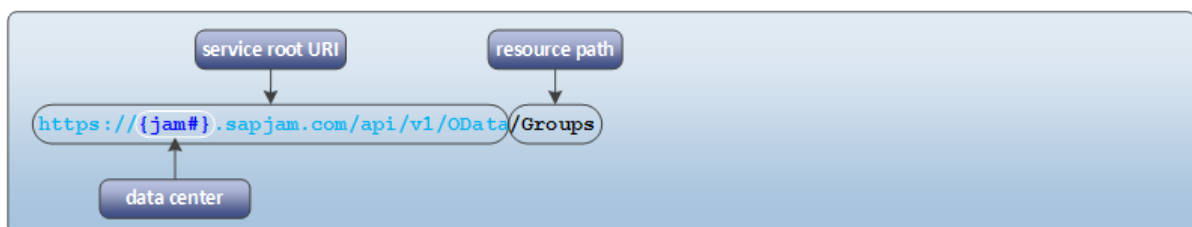
- **Slug: {file\_name\_without\_extension}** "Slug" is an HTTP header used in the SAP Jam Collaboration OData API as the name of the file used in SAP Jam. In an upload (a POST), the Slug header would be set as "profilePhoto" if that is the name you want the resource to appear as in SAP Jam. In an update (a PATCH) or a download (a GET), the Slug header would be set as "profilePhoto" if you want to update (re-upload) or download a file of that name in SAP Jam Collaboration. For example:

```
Slug: profilePhoto
```

## 8.1.2.2.5 URLs and URL parameters

OData APIs are accessed via a URL that calls a specific endpoint and has the required URL parameters properly set. There are different forms of URL for each of several specific types of operation.

The most basic API call specifies an entity, but not a specific resource. This is done for POST operations where a new resource has not yet been assigned a unique ID by SAP Jam, and in collection-valued GET requests, from which you want to retrieve a collection (or feed) of the available resources of that entity type. The basic parts of such a simple request (get me the available groups) are:



## service root

Identifies the data center or server of an OData service and location of the API in that data center or server. A generalized form of the service root URI is shown for the SAP Jam OData API.

## resource path

Identifies the resource to be interacted with. In the preceding diagram, the resources involved are one or more ContentItems in the SAP Jam Group that is specified by its unique ID.

Note that the combination of the HTTP method and a generalized form of the resource path is used as the signature of the endpoints in the SAP Jam OData API.

## {jam#} usage in the URLs (data centers)

SAP Jam Collaboration services are located in data centers in various locations around the world. Your organization's SAP Jam service will be at one of these locations, which must appear in the URL of your API calls, and which is indicated in this documentation by {jam#}, which is always in the form "jam#", in which "#" represents the 1 or 2 digit number of the data center in which your organization's SAP Jam instance is hosted. View the URL of any of your organization's SAP Jam pages to find your SAP Jam data center number.

## Unique IDs

To identify a specific resource, the SAP Jam OData API uses 22-character unique identifiers for the various accessible objects in SAP Jam Collaboration. This is shown in the following URL diagram:



For most entities, the unique ID alone is sufficient to identify a specific resource, but for some entities two attribute-value pairs are required. These one or two properties are commonly termed the "key" for the resource, and the key value, or values, must be set in the URL of an API call to indicate exactly which resource to perform the API call operation on. The entities that require two attribute-value pairs to identify a specific resource, and the attributes that they use are as follows:

- `ContentItem(Id='{Id}',ContentItemType='{ContentItemType}')`, where the '{ContentItemType}' can be "Page", "BlogEntry", "Document", "Tool", or "Poll".
- `ContentListItem(Id='{Id}',ContentListItemType='{ContentListItemType}')`, where the '{ContentListItemType}' can be "Page", "BlogEntry", "Document", "Tool", or "Poll".

- `EventResponse (RepondentId=' {RespondentId} ',EventId=' {EventId} ')`
- `Folder (Id=' {Id} ',FolderType=' {FolderType} ')`, where the `{FolderType}` can be "Folder" or "PrivateFolder".
- `GroupExternalObject (GroupId=' {GroupId} ',ExternalObjectId=' {ExternalObjectId} ')`
- `GroupMembership (GroupId=' {GroupId} ',MemberId=' {MemberId} ')`
- `GroupTemplate (Id=' {Id} ',GroupTemplateType=' {GroupTemplateType} ')`, where the `{GroupTemplateType}` can be "system" or "custom".
- `Kudo (Id=' {Id} ',KudoType=' {KudoType} ')`, where the `{KudoType}` can be "system" or "custom".
- `ObjectReference (Id=' {Id} ',Type=' {Type} ')`, where the `{Type}` can be "Member", "Group", "WallComment", "Event", "Task", "MemberKudo", "Comment", "FeedEntry", "ForumItem", or "ContentItem".
- `TaskAssignment (AssigneeId=' {AssigneeId} ',TaskId=' {TaskId} ')`
- `ThumbnailImage (Id=' {Id} ',ThumbnailImageType=' {ThumbnailImageType} ')`, where currently the only valid value for `{ThumbnailImageType}` is "48x48".
- `ThumbnailKudoImage (Id=' {Id} ',ThumbnailImageType=' {ThumbnailImageType} ')`, where currently the only valid value for `{ThumbnailImageType}` is "48x48".

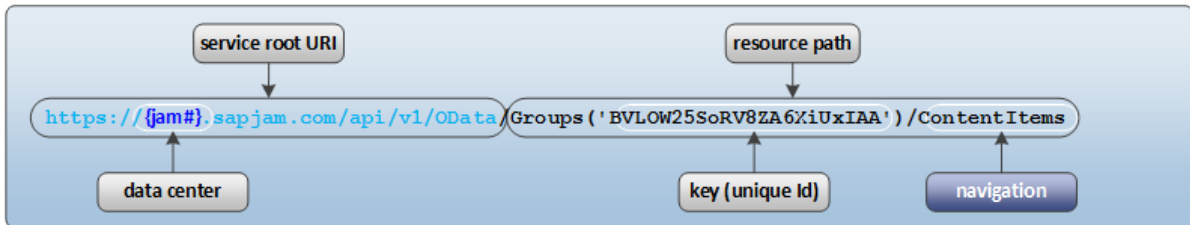
For the specification reference, see <http://www.odata.org/documentation/odata-version-3-0/common-schema-definition-language-csdl/#csdl6.2>.

## navigations

Navigations are URL extensions from the base entity to closely related entities. They can be named according to their "role", for example, one navigation from the Group entity is to a "Creator", which is the role, but which returns the "Member" information on the individual who created the group. Every entity has a predefined set of navigations. For example, the Group entity has the following navigations:

- Role: AllContentItems (Entity: ContentItem)
- Role: AllDiscussions (Entity: Discussion)
- Role: AllFolders (Entity: Folder)
- Role: AllIdeas (Entity: Idea)
- Role: AllQuestions (Entity: Question)
- Role: ContentItems (Entity: ContentItem)
- Role: ContentListItems (Entity: ContentListItem)
- Role: Creator (Entity: Member)
- Role: FeaturedExternalObjects (Entity: ExternalObject)
- Role: FeedEntries (Entity: FeedEntry)
- Role: Folders (Entity: Folder)
- Role: Forums (Entity: Forum)
- Role: Memberships (Entity: GroupMembership)
- Role: ParentGroup (Entity: Group)
- Role: PrimaryExternalObject (Entity: ExternalObject)
- Role: PrimaryGadgetObject (Entity: GroupGadgetObject)
- Role: ProfilePhoto (Entity: Image)
- Role: SubGroups (Entity: Group)

- Role: Tasks (Entity: Task)
- Role: Template (Entity: GroupTemplate)
- Role: UpcomingEvents (Entity: Event)



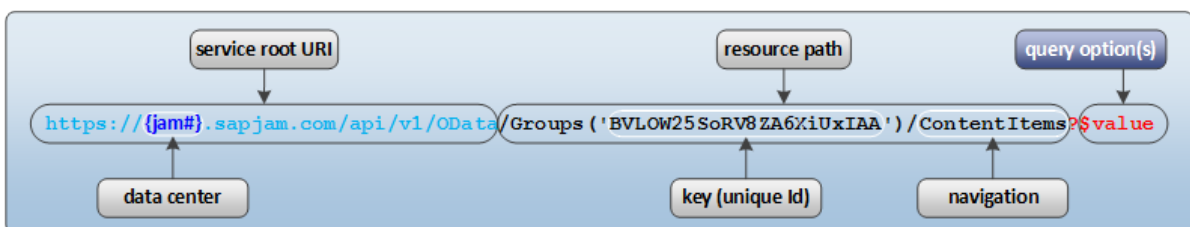
## query options

Query options are URL parameters that are added to the end of the URL.

There are three general types of query options:

- **System query options:** are indicated with a preceding dollar sign "\$". In the diagram above, the \$value segment is a system query option.
- **Custom query options:** are not used in the SAP Jam OData API, and so they are not discussed any further here.
- **Service operation parameters:** are the required parameters for service operations, which are discussed at greater length below.

The SAP Jam Collaboration OData API supports OData system query options with a number of limitations to ensure system responsiveness.



- There can be at most one Collection-valued navigation in an SAP Jam OData request.
- There can be at most 3 \$expand operations in one API call. Using \$expand on a Collection valued navigation of a Collection is not allowed.  
See in "OData Query Parameters", Use \$expand to include details on any property for which there are available navigations.
- Using \$filter is currently only supported for select properties of the following entities:
  - **ContentItem:** ContentItemType
  - **ContentListItem:** Name
  - **ExternalObject:** Exid, ObjectType
  - **Folder:** Name
  - **FeedEntry:** Read
  - **ForumItem:** ForumItemType
  - **Group:** Id, Name, IsActive, GroupType

- **GroupExternalObject:** LinkType
- **GroupMembership:** MemberType
- **Idea:** Status
- **Notification:** Category, EventType
- **Question:** HasBestAnswer
- **Task:** IsOverdue
- **TaskAssignment:** Status

See in "OData Query Parameters", Use `$filter` to limit the entries returned according to their content.

- SAP Jam will never return arbitrarily large collections. Either server-driven paging (use `$skiptoken`) or client-driven paging (use `$skip`) will apply. Certain endpoints only support server-driven paging (`$skiptoken`). In general, a maximum of 20 items will be returned, unless a value of `$top` is specified (and supported), in which case the maximum value is 100.

See in "OData Query Parameters":

- Use `$skip` to offset the set of entries returned.
- Use `$skiptoken` to offset the set of entries returned.
- Use `$top` to set the number of entries returned.

- For FeedEntries and Notifications API calls, `$skip`, `$top`, and `$count` are not available. Also, `$count` is not available for GroupMembership API calls, and `$orderby` is not available for FeedEntries and Notifications, but it will be selectively available where the items are sortable in the SAP Jam user interface.

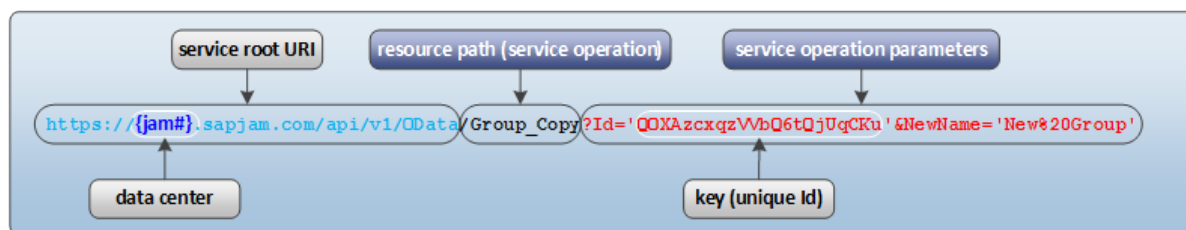
See in "OData Query Parameters":

- Use `$skip` to offset the set of entries returned.
- Use `$top` to set the number of entries returned.
- Use `$count` to discover how many entries there are.
- Use `$orderby` to set the criteria that the returned entries are ordered by.

See the OData v.2 Specification's discussion of [OData System Query Options](#) .

## Service Operations

In practice, a service operation is any endpoint that performs a required task that does not fit with the usual POST, GET, PATCH, or DELETE operations. In SAP Jam Collaboration's implementation, service operations are signified by using underscores between words in the endpoint name rather than showing the name in camel-case. There are, however, at least four exceptions: `[GET] /Self`, `[GET] /Company`, `[GET] /Search`, and `[GET] /SearchSummary`.

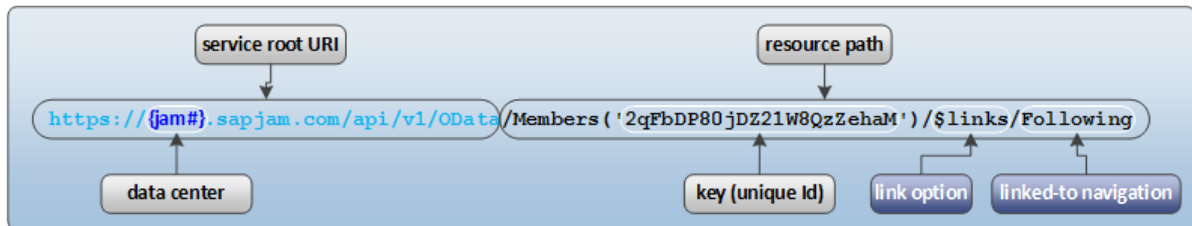


Note that if you don't find sufficient information about a service operation in this *SAP Jam Collaboration API Tutorial* or the *SAP Jam Collaboration API Reference*, then you can check the `$metadata` file. Each service operation is described to some extent in that file.

For the technical specification for a service operation, see section 2.13 of the OData 2.0's [Operations](#) page.

## Links between entries

The OData specification allows for the creation of links between entries.



For example, in the SAP Jam Collaboration OData API, currently logged-in Member can be set as "Following" another specified Member by setting a "\$link" from the Member to be followed to "Following", which will take the Id of the currently logged-in Member as the "follower". For example:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members('2qFbDP80jDZ21W8QzZehaM')/$links/Following
```

There are a small number of such operations available in the SAP Jam OData API, but they are important operations. They include:

- [POST] /Members('{id}')/\$links/Following
- [DELETE] /Members('{id}')/\$links/Following('{id1}')
- [POST] /Questions('{id}')/\$links/BestAnswer
- [DELETE] /Questions('{id}')/\$links/BestAnswer
- [POST] /Groups('{id}')/\$links/FeaturedExternalObjects
- [DELETE] /Groups('{id}')/\$links/FeaturedExternalObjects('{id1}')
- [POST] /Events('{id}')/\$links/Invitees
- [POST] /Tasks('{id}')/\$links/Attachments
- [DELETE] /Tasks('{id}')/\$links/Attachments('{id1}')
- [POST] /Tasks('{id}')/\$links/PendingFollowers

See sections [2.9 Manipulating Links](#), [2.10 Creating Links between Entries](#), [2.11 Removing Links between Entries](#), and [2.12 Replacing Links between Entries](#) in the Operations page of the OData 2.0 specification.

### 8.1.2.2.6 Response payload data

GET and POST calls will return a payload of full resource information in the format requested.

The response payload can be very verbose or very concise, depending on both the entity type returned and the system query options, if any, set in the API call. As an example, the XML response for a GET for a single Group with an expansion of the Creator set—`[GET] /Group('{Id}')?$expand=Creator`—is shown in the following code block. Note that the original version of this segment was 189 (pretty-printed) lines long before 31

navigation lines were removed. (Non-pretty-printed, this example would be a single line, even with all of the navigations left in, as the return has no line breaks or indenting.)

```
<?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">
  <id>https://<jam#> [page 246].sapjam.com/api/v1/OData/
Groups ('RsOpnZp4b12KRc5KR8vY4Q')</id>
  <title type="text">Groups ('RsOpnZp4b12KRc5KR8vY4Q')</title>
  <updated>2016-08-30T13:59:12+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Groups ('RsOpnZp4b12KRc5KR8vY4Q') "
    href="Groups ('RsOpnZp4b12KRc5KR8vY4Q')"/>
  <!-- Four Groups navigations removed from here. -->
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Creator"
    type="application/atom+xml;type=entry" title="Creator"
    href="Groups ('RsOpnZp4b12KRc5KR8vY4Q')/Creator">
  <m:inline type="application/atom+xml;type=entry">
    <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">
      <id>https://<jam#> [page 246].sapjam.com/api/v1/OData/
Members ('3carMxYCZJP2mskM4MR5OP')</id>
      <title type="text">Members ('3carMxYCZJP2mskM4MR5OP')</title>
      <updated>2016-08-30T13:59:12+00:00</updated>
      <author>
        <name/>
      </author>
      <link rel="edit" title="Members ('3carMxYCZJP2mskM4MR5OP') "
        href="Members ('3carMxYCZJP2mskM4MR5OP')"/>
      <!-- 11 Member navigations removed from here. -->
      <category term="SAPJam.Member"
        scheme="http://schemas.microsoft.com/ado/2007/08/
dataservices/scheme"/>
      <content type="application/xml">
        <m:properties>
          <d:Id m:type="Edm.String">3carMxYCZJP2mskM4MR5OP</d:Id>
          <d:FirstName m:type="Edm.String">John</d:FirstName>
          <d:LastName m:type="Edm.String">Does</d:LastName>
          <d:Nickname m:type="Edm.String" m:null="true"/>
          <d>Title m:type="Edm.String">Software Developer</d>Title>
          <d>Email m:type="Edm.String">john.doe@example.com</
d>Email>
          <d:FullName m:type="Edm.String">John Doe</d:FullName>
          <d:Role m:type="Edm.String">company</d:Role>
          <d:IsFollowing m:type="Edm.Boolean">>false</d:IsFollowing>
          <d:WebURL m:type="Edm.String"
            >https://<jam#> [page 246].sapjam.com/profile/wall/
3carMxYCZJP2mskM4MR5OP</d:WebURL>
          <d:IsAway m:type="Edm.Boolean">>false</d:IsAway>
        </m:properties>
      </content>
    </entry>
  </m:inline>
</link>
  <!-- 16 more Groups navigations removed from here. -->
  <category term="SAPJam.Group"
    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
  <content type="application/xml">
    <m:properties>
      <d:Id m:type="Edm.String">RsOpnZp4b12KRc5KR8vY4Q</d:Id>
```

```

    <d:Name m:type="Edm.String">Customer Requests Repercussion Group</
d:Name>
    <d:Description m:type="Edm.String"/>
    <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
    <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
    <d:Announcement m:type="Edm.String" m:null="true"/>
    <d:OverviewAsLanding m:type="Edm.Boolean">true</d:OverviewAsLanding>
    <d:Participation m:type="Edm.String">full</d:Participation>
    <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
    <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
    <d:ModerationPolicy m:type="Edm.Boolean">false</d:ModerationPolicy>
    <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
    <d:CreatedAt m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</
d:CreatedAt>
    <d:LastModifiedAt m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</
d:LastModifiedAt>
    <d:LastActivityAt m:type="Edm.DateTimeOffset">2016-06-20T17:40:32Z</
d:LastActivityAt>
    <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
    <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
    <d:DisableAtNotify m:type="Edm.Boolean">false</d:DisableAtNotify>
    <d:TermsOfUse m:type="Edm.String" m:null="true"/>
    <d:WebURL m:type="Edm.String"
    >https://<jam#> [page 246].sapjam.com/groups/
RsOpnZp4b12KRc5KR8vY4Q</d:WebURL>
    <d:ContentsVisible m:type="Edm.Boolean">true</d:ContentsVisible>
    <d:QuestionsVisible m:type="Edm.Boolean">true</d:QuestionsVisible>
    <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>
    <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
    <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
    <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
    <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
    <d:SubgroupsVisible m:type="Edm.Boolean">true</d:SubgroupsVisible>
    <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
    <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
    <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
    <d:IsAdmin m:type="Edm.Boolean">true</d:IsAdmin>
    <d:EmailFrequency m:type="Edm.String">none</d:EmailFrequency>
  </m:properties>
</content>
</entry>

```

There are a few significant parts in these responses worthy of comment here:

- This example's outer wrapping tags are `<entry>` `</entry>` tags, which indicates that this is the return of a single resource record. If there were a series of entries, wrapped in `<feed>` `</feed>` tags, this would be the return of a collection of resources.
- The lines that begin with "`<link rel="http://schemas.microsoft.com/ado ...`" are the navigations available from a Group entity. They show the resource path for a GET or POST call to the navigation, and they indicate the type of response you would get from a GET call (feed or entry) for each navigation. All of the navigation information has been removed to keep this example concise, except for the "Creator" navigation, which is retained to show better how the expansion system query option works. The removal of the navigations is the practice throughout this tutorial.
- The resource's properties are listed towards the bottom of the response payload, wrapped appropriately in `<m:properties>` `</m:properties>` tags.
- The properties show the property name, its data type, and its value.
- The data type is prefixed with "Edm.", which refers to the Entity Data Model, which is described in the next section of this page.



As mentioned, the JSON responses are considerably more concise. They are also arranged somewhat differently, and they lack the wrapping tags called out in the preceding list. They also lack some of the elements described above (such as the "Edm." prefixes). There is little benefit in adding a JSON example here, however, and you will very quickly discover the differences as you read through this tutorial.

## Entity Data Model

Data in the SAP Jam Collaboration OData API is represented in accordance with the Entity Data Model (EDM). The Entity Data Model specifies a set of data types (for example, `Edm.Int32`, `Edm.String`, `Edm.Boolean`, and `Edm.DateTimeOffset`) be used to describe a resource.

For more information on the Entity Data Model, see:

- [A general description of the Entity Data Model](#) ➤ .
- [The full technical details of the Entity Data Model](#) ➤ .

## Generated Properties

Many properties, such as unique Ids, timestamps, counts, and URLs, are generated by the SAP Jam Collaboration service. Some of these properties are structural aspects of SAP Jam, others report on member actions, and others show manually selected system configuration options. For example, the "LikesCount" on a Comment is the total number of "Likes" set by the members who have viewed the comment; this value is a sum of those "Like" actions that is generated by the SAP Jam system, and which normally cannot be set manually in a POST or PATCH operation. Generally, no "generated property" can be set via the API.

### 8.1.2.2.7 HTTP success codes

Successful SAP Jam Collaboration API requests receive the "success code" that is appropriate for the given type of request.

The successful requests are:

- For a [GET], **200 OK**: The requested resource retrieval is successful, and a full payload of the requested resource is returned.
- For a [PATCH], **204 No Content**: The requested resource patch is successful; no payload is returned.
- For a [POST] that creates a resource, **201 Created**: The requested resource creation is successful, and a full payload of the created resource is returned. The location of the created resource is included in the returned Location HTTP header.
- For a [POST] that does not create a resource, **204 No Content**: The requested resource post is successful; as no content is created, no payload is returned.
- For a [DELETE], **204 No Content**: The requested resource deletion is successful; no payload is returned.

## 8.1.2.2.8 HTTP error codes

Unsuccessful SAP Jam Collaboration API requests receive an "error code" that is appropriate for the given type of request and the nature of the problem.

Requests that fail or encounter problems are typically:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource. SAP Jam provides greater detail on this particular error message to provide a clearer indication of the problem.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

### i Note

Clients should gracefully handle error responses. For example, if SAP Jam is under maintenance, or the API quota ([rate limits \[page 254\]](#)) is exhausted, an error can occur. Any API call can produce an error result.

### i Note

Detailed error message text is subject to change and should not be relied upon or parsed.

## Rate Limits

While not specifically an HTTP feature, rate limits are imposed by the SAP Jam Collaboration service per member per time span to ensure server responsiveness. These requests are measured in units. There is a burst limit of 200 units per minute, every minute, and there is an hourly limit of 800 units per hour. Note that these rate limits are subject to change without notice, but they are set to balance server responsiveness with substantial use of API calls. The explanation of rate limits is included here because violation of those limits will trigger a `429 Too Many Requests` HTTP response.

## 8.1.2.2.9 Understanding the OData \$metadata file

The OData \$metadata file is a CSDL file that is made available to clients to help them discover an OData API's structure and organization.

### CSDL

The Conceptual Schema Definition Language (CSDL) is an XML-based language that describes the entities, relationships, and functions that make up the conceptual model of a data-driven application.

- [CSDL v3 Specification](#) 

### OData \$metadata

While the \$metadata file is primarily intended for client application use, it does provide human-readable information on the remote application's API. The sections of information that provides are as follows:

- **EntityType:** Describes each entity, its properties (including information on its data type, and whether it is nullable in POSTs, filterable or sortable in collection-valued GETs, and whether it is updatable [can be PATCHed]), and its navigations.
- **Association:** Provides details on each navigation available in the API, including its role, type, and multiplicity. (Role can be "Creator" while type is "Member", for example; multiplicity indicates the number of resources that can be involved at each "end" of the association or navigation.)
- **EntitySet:** Provides a concise listing of all the entities available in the API, with information on their name, entity type, and whether they are creatable, updatable, or deletable.
- **AssociationSet:** Provides further details on navigations, including whether each navigation is creatable, updatable, or deletable.
- **FunctionImport (Service Operations):** Provides information on each service operation available in the API, including their name, HTTP method, the entity that they act on, the name and data type of any required parameters and the type of data that they return, if any (such as data type, entity type, and whether the return is a collection).

Note that this information is what informs the entity diagrams at the beginning of each "examples type" lesson, and provides much of the information shown in the various property lists and tables used throughout this tutorial. Once you are familiar with the SAP Jam OData API, you will not have any need for the \$metadata file information or the diagrams, tables, and lists in this tutorial, as this information is also shown in the SAP Jam OData API Reference.

## 8.1.2.2.10 Use of curl commands

This tutorial uses curl commands to summarize the information that must be passed to make an effective API call. A **curl command** (sometimes pronounced "see-URL") provides a concise view of exactly what elements

have to be passed in the API call. This is most helpful if you are using `lib-curl` for your encoded API calls, although other options are available.

The curl command options are always shown in the same order so that you can quickly evaluate what the different requirements are for each call. These options and their order are:

- `-i` — (include [the HTTP header]) **[curl only]**: includes the HTTP responses, some server information, and some request and response data in the response. (Note that none of this information is shown in any of the examples in this tutorial.)
- `-H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"` — (header) **[Required]**: the SAP Jam OData API requires either OAuth 1.0a or 2.0 authentication to respond to an API call.
- `-H "Content-Type: application/{atom+xml|json}"` — (header) **[Required for POST and PATCH requests]**: indicates the MIME type of the body of the request or of the file to be uploaded.
- `-H "Accept: application/{atom+xml|json}"` — (header) **[Optional]**: indicates the acceptable Content-Types for the response. OData APIs default to XML, so if you want JSON, you must either specify it in this header or set that as a query option in the URL.
- `-H "Stub: {file_name}"` — (header) **[Required for API calls that upload or download files (POSTs or PATCHes and GETs)]**: indicates the filename (less the extension) that will be used in SAP Jam for an uploaded file.
- `-X {POST|PATCH|DELETE}` — (request) **[Required for POSTs, PATCHes, and DELETEs]**: used to indicate some other HTTP method than GET, which is the default if this option is not specified.
- `"{the_API_call_URL}"` (URL) — **[Required]**: this must be the full URL of the API call.
- `-d` — (data) **[Required for POSTs and PATCHes]**: this is the data used to create (POST) or update (PATCH) a resource. The `-d` option is typically formed in one of the following ways:
  - `-d "{ \"Name\": \"Customer Requests Discussions Group\" }"` — a short bit of JSON data included in the curl command.
  - `-d "<Name>Customer Requests Discussion Group</Name>"` — a short bit of XML data included in the curl command.
  - `-d@C:{path_to_payload_file}/{payload_file_name}.json` — a longer bit of JSON content sent in a payload file.
  - `-d@C:{path_to_payload_file}/{payload_file_name}.xml` — a longer bit of XML content sent in a payload file.
- `-x <proxy.myorg.com:8080>` — (proxy) **Required only if your network is behind a web proxy**. Should show the host, subdomain, domain, and port of the proxy. The proxy option is not shown in any examples in this tutorial, but it may be required to work in your organization.

## 8.1.2.3 The Groups and related API

This lesson introduces you to the SAP Jam OData API's "Groups", "GroupTemplate", "SystemGroupTemplate", and "CustomGroupTemplate" entities and many of the API calls (or endpoints) that are used to create and manage Groups. This lesson also introduces you to the basic handling of OData POST, GET, PATCH, DELETE, and service operations. Groups are the fundamental organizational structure of SAP Jam Collaboration. All content, forums, integrated external application data, and OpenSocial Gadgets are available within specified groups, and all members draw their information from, and converse within, groups. All SAP Jam utilities and services, such as calendar events and tasks, are enabled for use in groups, and relate to that group's purpose and operations.

## Learning Objectives

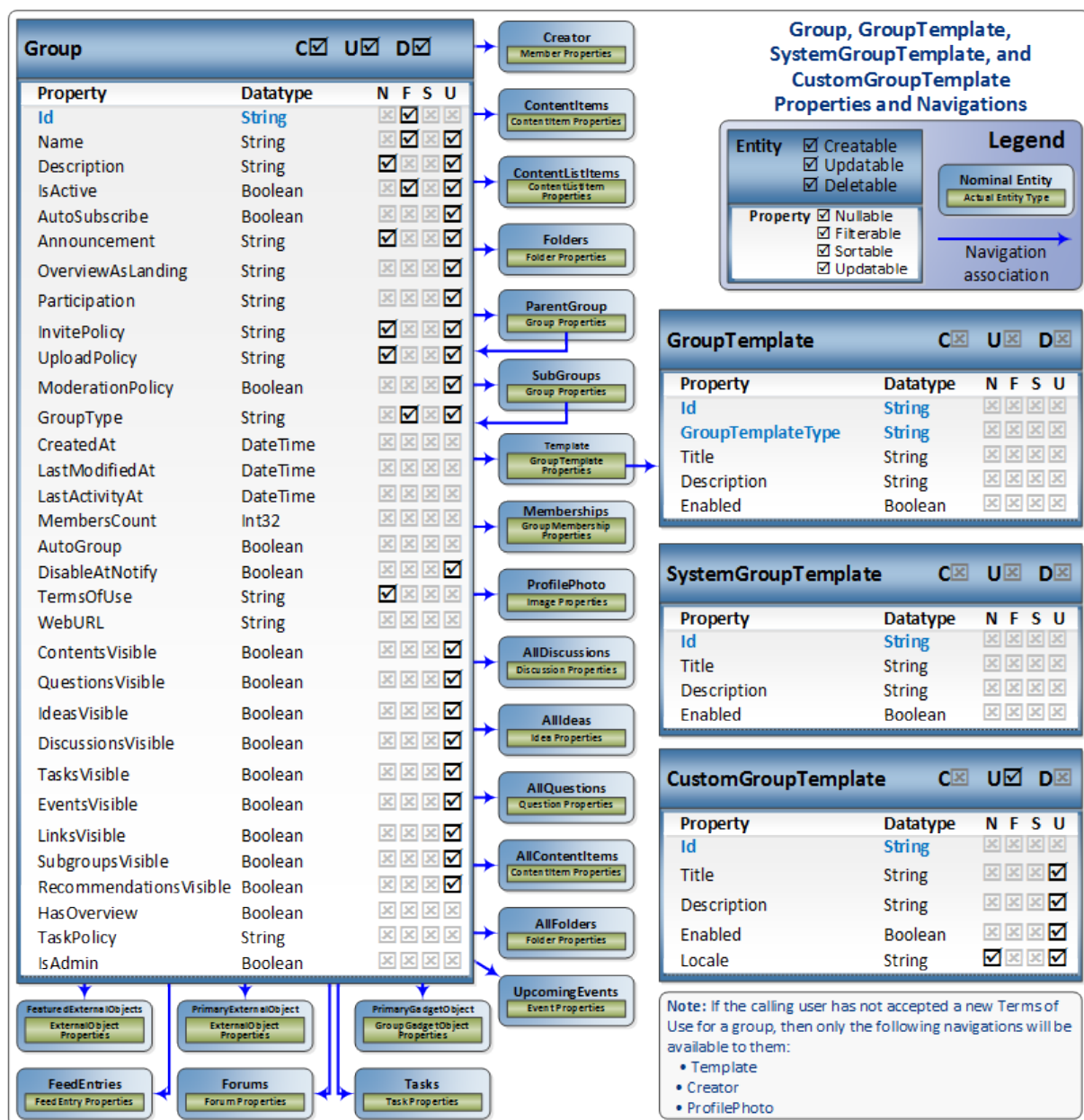
The learning objectives for this page are:

- Get an initial sense of the [properties and navigations \[page 257\]](#) that make up, and are available to, these entities.
- Get an initial sense of the [endpoints \[page 258\]](#) that are available to use and manage these entities.

## Group and related entities' properties and navigations

The following diagram is provided to give you a good sense of what data (or "properties") the depicted entities represent. The diagram also shows the [navigations \[page 247\]](#) from each of the entities covered. There are four OData entities that are discussed in this lesson of the tutorial and whose properties and navigations are depicted in the following diagram:

- **Groups:** are structured areas for collaboration on a particular topic area. Members are invited to join Groups, where they can discuss issues, ask questions, and present ideas. Members can also upload, view, and download documents, create wikis and blogs, and view a group calendar of events.
- **GroupTemplates:** are pre-set skeletal Group configuration that can be used as the basis for creating a Group. The GroupTemplates entity in particular is a combination of SystemGroupTemplates and CustomGroupTemplates.
- **SystemGroupTemplates:** are the group templates that are provided by SAP Jam.
- **CustomGroupTemplates:** are the customized group templates that have been created by users in your organization for use in your instance of SAP Jam.



Group, GroupTemplate, SystemGroupTemplate, and CustomGroupTemplate entities' properties and navigations

## Available Group and related entities' endpoints






The following list of API calls is shown only to give you, the new SAP Jam OData API user, a good sense of what operations are available for the Groups and the various GroupTemplates entities.

While there are many more Groups endpoints that those listed below, these are the ones that are relevant to this basic introduction to the Groups and related entities:

### Note

In the list below, the endpoint signatures link to the SAP Jam Collaboration API Reference. The "(See ...)" notes at the end of the endpoint descriptions link to pages that cover the endpoint in this tutorial.

- [\[POST\] /Groups](#)  — Creates a group. (See [Create a group with minimal settings](#) [page 260], [Create a group with all options set](#) [page 266], and [Create a Group from a template](#) [page 294].)
- [\[GET\] /Groups](#)  — Retrieves a collection of information on all Groups that the current Member can access. (See [Retrieve a feed of available GroupTemplates](#) [page 289].)
- [\[GET\] /Groups\('{id}'\)](#)  — Retrieves the information on the specified Group. (See [Retrieve a group's information](#) [page 277].)
- [\[PATCH\] /Groups\('{id}'\)](#)  — Updates the information on the specified Group. (See [Update an existing Group](#) [page 304].)
- [\[DELETE\] /Groups\('{id}'\)](#)  — Deletes the specified Group. (See [Delete an existing Group](#) [page 308].)
- [\[GET\] /Groups\('{id}'\)/ParentGroup](#)  — Retrieves the information on the Parent Group of the specified Group.
- [\[POST\] /Groups\('{id}'\)/SubGroups](#)  — Creates a SubGroup of the specified Group.
- [\[GET\] /Groups\('{id}'\)/SubGroups](#)  — Retrieves a collection of information on the SubGroups of the specified Group.
- [\[GET\] /Groups\('{id}'\)/Template](#)  — Retrieves the information on the GroupTemplate that was used to create the specified Group.
- [\[GET\] /Groups\\_Autocomplete](#)  — Performs a string-based search on Groups that is restricted to the Groups the current logged in Member can access and returns a collection of the Groups that match the specified Query.
  - Parameter: **Query**: The string that the search will be based on. (See [Search for a Group](#) [page 299].)
- [\[POST\] /Group\\_Copy](#)  — Copies a Group.
  - Parameter: **Id**: The Id of the Group to be copied.
  - Parameter: **NewName**: The new, unique name for the Group that is to be created. (See [Copy an existing group](#) [page 273].)
- [\[GET\] /Group\\_FeedLatestCount](#)  — Retrieves a count of the FeedEntries posted to a Group's wall ("Feed Updates" section) since the specified last known FeedEntry Id.
  - Parameter: **Id**: The Id of the Group for which you want to retrieve a count of the latest FeedEntries.
  - Parameter: **LatestTopLevelId**: The Id of the last known top-level FeedEntry.
- [\[GET\] /Groups\\_IAdminister](#)  — Retrieves a collection of Groups information of the Groups for which the currently logged-in user is an administrator.
- [\[POST\] /Group\\_SaveAsTemplate](#)  — Saves the structure and content of the indicated Group as a CustomGroupTemplate for future use in setting up a new, similar Group. This will include Forums and blogs, but will exclude FeedEntries, ContentItems (except for blogs), and any Folders structure.
  - Parameter: **Id**: The Id of the Group that you want to save as a CustomGroupTemplate.
  - Parameter: **Title**: The title or name that you want to give to the created CustomGroupTemplate.
  - Parameter: **Description**: The description that you want to give to the created CustomGroupTemplate.
  - Parameter: **Enabled**: Whether you want the created CustomGroupTemplate to be visible and available to Members to create new Groups from. (See [Save a group as a CustomGroupTemplate](#) [page 282].)
- [\[GET\] /GroupTemplates](#)  — Retrieves a collection of all GroupTemplates that are visible to the current user. Note that "GroupTemplates" are a combination of SystemGroupTemplates and CustomGroupTemplates. (See [Retrieve a feed of available GroupTemplates](#) [page 289].)
- [\[GET\] /GroupTemplates\(Id='{Id}', GroupTemplateType='{GroupTemplateType}'\)](#)  — Retrieves the information on the specified GroupTemplate.

- [\[GET\] /SystemGroupTemplates](#)  — Retrieves a collection of all SystemGroupTemplates that are visible to the currently logged-in user.
- [\[GET\] /SystemGroupTemplates\('{id}'\)](#)  — Retrieves the information on the specified SystemGroupTemplate.
- [\[GET\] /CustomGroupTemplates](#)  — Retrieves a collection of all CustomGroupTemplates that are visible to the currently logged-in user.
- [\[GET\] /CustomGroupTemplates\('{id}'\)](#)  — Retrieves the information on the specified CustomGroupTemplate.
- [\[PATCH\] /CustomGroupTemplates\('{id}'\)](#)  — Updates the information on the specified CustomGroupTemplate. (See [Update a CustomGroupTemplate \[page 286\]](#).)

### Note

Only the endpoints above with a "(See ...)" note at the end of the endpoint description are discussed in this tutorial. There should be sufficient examples shown that the use of other endpoints can be easily understood. Also, there are many Groups endpoints that are discussed in other lessons in this tutorial.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you have an initial sense of the [properties and navigations \[page 257\]](#) that make up, and are available to, these entities?
- Do you have an initial sense of the [endpoints \[page 258\]](#) that are available to use and manage these entities?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Create a group with minimal settings \[page 260\]](#).

### 8.1.2.3.1 Create a group with minimal settings

This page of "The Groups and related API" shows you how to create a Group with the minimal information required by using the `[POST] /Groups` endpoint. This is the first explanation of how to make a POST call using the SAP Jam OData API.

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [a minimal request to create a group \[page 261\]](#).
- Know what the [possible responses \[page 261\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a POST call \[page 265\]](#) so that you can easily perform other POST operations.



## Develop the POST request to create a group

This section shows the minimal information that is required to create a group.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "POST" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups
```

Where the variable URL parameter for this call is:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
5. Set the name of the new group that you want to create in the form "Name": "Customer Requests Discussions Group", where "Customer Requests Discussions Group" should be set to whatever name the user chooses to give to the new group.

To summarize these requirements of the request to create a group by providing the minimum required information, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
-H "Content-Type: application/json" -H "Accept: application/json"  
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"  
-d "{ \"Name\": \"Customer Requests Discussion Group\" }"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"  
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"  
-d "<Name>Customer Requests Discussion Group</Name>"
```

## Develop the ability to handle either the success or the error response

If your POST request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that creates a resource, such as this request, the response code is:

```
201 Created
```

- For any POST operation that creates a resource, you also receive a payload of the full information about the created resource.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{"d": {"results": {
```

```

    "__metadata": {
      "uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')",
      "type": "SAPJam.Group"
    },
    "Id": "QOXAzcqxzVVbQ6tQjUqCKu",
    "Name": "Customer Requests Discussion Group",
    "Description": "",
    "IsActive": true,
    "AutoSubscribe": false,
    "Announcement": null,
    "OverviewAsLanding": true,
    "Participation": "full",
    "InvitePolicy": "all",
    "UploadPolicy": "all",
    "ModerationPolicy": false,
    "GroupType": "public internal",
    "CreatedAt": "/Date(1466446118000)/",
    "LastModifiedAt": "/Date(1466446118000)/",
    "LastActivityAt": "/Date(1466446118000)/",
    "MembersCount": 1,
    "AutoGroup": false,
    "DisableAtNotify": false,
    "TermsOfUse": null,
    "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
QOXAzcqxzVVbQ6tQjUqCKu",
    "ContentsVisible": true,
    "QuestionsVisible": true,
    "IdeasVisible": true,
    "DiscussionsVisible": true,
    "TasksVisible": true,
    "EventsVisible": true,
    "LinksVisible": true,
    "SubgroupsVisible": true,
    "RecommendationsVisible": true,
    "HasOverview": false,
    "TaskPolicy": "readonly",
    "IsAdmin": true,
    "PrimaryGadgetObject": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/PrimaryGadgetObject"}},
    "FeedEntries": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
FeedEntries"}},
    "ParentGroup": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
ParentGroup"}},
    "SubGroups": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
SubGroups"}},
    "Creator": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Creator"}},
    "ProfilePhoto": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
ProfilePhoto"}},
    "ContentItems": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
ContentItems"}},
    "AllContentItems": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/AllContentItems"}},
    "Folders": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Folders"}},
    "AllFolders": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
AllFolders"}},
    "ContentListItems": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/ContentListItems"}},
    "Forums": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Forums"}},
    "Memberships": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Memberships"}},
    "FeaturedExternalObjects": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/FeaturedExternalObjects"}},
    "PrimaryExternalObject": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/PrimaryExternalObject"}},

```

```

    "AllDiscussions": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/AllDiscussions"}}},
    "AllIdeas": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
AllIdeas"}}},
    "AllQuestions": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
AllQuestions"}}},
    "Template": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Template"}}},
    "Tasks": {"__deferred": {"uri": "Groups('QOXAzcqxzVVbQ6tQjUqCKu')/
Tasks"}}},
    "UpcomingEvents": {"__deferred": {"uri":
"Groups('QOXAzcqxzVVbQ6tQjUqCKu')/UpcomingEvents"}}
  }}}

```

The many navigation URIs for the created group are retained in the response example above. In most future examples, these will be cut out to keep the code examples concise. Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/odata/
Groups('RsOpnZp4b12KRc5KR8vY4Q')</id>
  <title type="text">Groups('RsOpnZp4b12KRc5KR8vY4Q')</title>
  <updated>2016-06-20T17:40:33+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Groups('RsOpnZp4b12KRc5KR8vY4Q') "
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
PrimaryGadgetObject" type="application/atom+xml;type=entry"
title="PrimaryGadgetObject" href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/
PrimaryGadgetObject"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
FeedEntries" type="application/atom+xml;type=feed" title="FeedEntries"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/FeedEntries"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
ParentGroup" type="application/atom+xml;type=entry" title="ParentGroup"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/ParentGroup"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
SubGroups" type="application/atom+xml;type=feed" title="SubGroups"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/SubGroups"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Creator" type="application/atom+xml;type=entry" title="Creator"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Creator"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
ProfilePhoto" type="application/atom+xml;type=entry" title="ProfilePhoto"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/ProfilePhoto"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
ContentItems" type="application/atom+xml;type=feed" title="ContentItems"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/ContentItems"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AllContentItems" type="application/atom+xml;type=feed"
title="AllContentItems" href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/
AllContentItems"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Folders" type="application/atom+xml;type=feed" title="Folders"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Folders"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AllFolders" type="application/atom+xml;type=feed" title="AllFolders"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/AllFolders"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
ContentListItems" type="application/atom+xml;type=feed"

```

```

title="ContentListItems" href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/
ContentListItems"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Forums" type="application/atom+xml;type=feed" title="Forums"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Forums"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Memberships" type="application/atom+xml;type=feed" title="Memberships"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Memberships"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
FeaturedExternalObjects" type="application/atom+xml;type=feed"
title="FeaturedExternalObjects" href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/
FeaturedExternalObjects"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
PrimaryExternalObject" type="application/atom+xml;type=entry"
title="PrimaryExternalObject" href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/
PrimaryExternalObject"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AllDiscussions" type="application/atom+xml;type=feed" title="AllDiscussions"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/AllDiscussions"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AllIdeas" type="application/atom+xml;type=feed" title="AllIdeas"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/AllIdeas"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AllQuestions" type="application/atom+xml;type=feed" title="AllQuestions"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/AllQuestions"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Template" type="application/atom+xml;type=entry" title="Template"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Template"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Tasks" type="application/atom+xml;type=feed" title="Tasks"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/Tasks"/>
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
UpcomingEvents" type="application/atom+xml;type=feed" title="UpcomingEvents"
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')/UpcomingEvents"/>
<category term="SAPJam.Group" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
<content type="application/atom+xml">
  <m:properties>
    <d:Id m:type="Edm.String">RsOpnZp4b12KRc5KR8vY4Q</d:Id>
    <d:Name m:type="Edm.String">Customer Requests Discussion Group</
d:Name>
    <d:Description m:type="Edm.String"/>
    <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
    <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
    <d:Announcement m:type="Edm.String" m:null="true"/>
    <d:OverviewAsLanding m:type="Edm.Boolean">true</
d:OverviewAsLanding>
    <d:Participation m:type="Edm.String">full</d:Participation>
    <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
    <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
    <d:ModerationPolicy m:type="Edm.Boolean">false</
d:ModerationPolicy>
    <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
    <d:CreatedAt m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</
d:CreatedAt>
    <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</d:LastModifiedAt>
    <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-06-20T17:40:32Z</d:LastActivityAt>
    <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
    <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
    <d:DisableAtNotify m:type="Edm.Boolean">false</d:DisableAtNotify>
    <d:TermsOfUse m:type="Edm.String" m:null="true"/>
    <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/RsOpnZp4b12KRc5KR8vY4Q</d:WebURL>
    <d:ContentsVisible m:type="Edm.Boolean">true</d:ContentsVisible>
    <d:QuestionsVisible m:type="Edm.Boolean">true</d:QuestionsVisible>
    <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>

```

```

        <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">true</d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">true</d:IsAdmin>
    </m:properties>
</content>
</entry>

```

### Note

The many navigation URIs for the created group are retained in the response example above. In most future examples, these will be cut out to keep the code examples concise.

If your POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About POST calls

The unique aspects of using POST calls are:

- You do not set the 'Id' of a resource as a URL parameter.
- You do not use query parameters, except for uploading content, as is shown in the later lesson, [Uploading and downloading files \[page 359\]](#).
- You must include a payload that specifies the properties of the resource that you want to create.
- The success code for a call that creates a resource is 201 `Created`.
- For a call that creates a resource, a payload listing all properties of the created resource is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to create a group \[page 261\]](#)?
- Do you know what the [possible responses \[page 261\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a POST call \[page 265\]](#) so that you can easily perform other POST operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Create a group with all options set \[page 266\]](#).

### 8.1.2.3.2 Create a group with all options set

This page of "The Groups and related API" shows you how to create a Group with all of the properties that you can set by using the `[POST] /Groups` endpoint. This is the second explanation of how to use a POST operation.

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to create a group setting most or all available options \[page 266\]](#).
- Know what the [possible responses \[page 270\]](#) are so that you can develop the ability to handle these responses.
- Understand the [unique aspects of a POST call \[page 272\]](#) so that you can easily perform other POST operations.

## Develop the POST request to create a group with all options set

This section shows the minimal information that is required to create a group.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "POST" request.

- Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups
```


Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- Set the name of the new group and any of the "optional" properties for which you want to set different values than the defaults. To do this, the request that you formulate must have an attached payload that lists all of the properties and their preferred values that you want to set.

The following table lists all of the properties that you can set, as well as their default values:

Group properties

Property	Type	Description	Creata- ble	N
<b>Name</b>	String	The Group's name, which must be unique within your organization's instance of SAP Jam. Maximum 255 characters.	Mandatory	
<b>Description</b>	String	The Group's description. Maximum 255 characters. <b>Default:</b> (no content).	Optional	
<b>IsActive</b>	Boolean	Whether the Group is set to be visible and usable to members. <b>Default:</b> "true".	Optional	
<b>AutoSubscribe</b>	Boolean	Whether the Group has been configured to have members automatically receive email notifications when new items are posted. <b>Default:</b> "false".	Optional	
<b>Announcement</b>	String	The message that members will see when they first visit the Group. <b>Default:</b> (no content).	Optional	
<b>OverviewAsLanding</b>	Boolean	Whether the Overview page has been configured to serve as the Group's landing page. <b>Default:</b> "false".	Optional	
<b>Participation</b>	String	The policy that determines who can participate in the Group. The valid values are "info" (read-only), "expert" (limited), or "full". <b>Default:</b> "full".	Optional	
<b>InvitePolicy</b>	String	The policy that determines who can send invitations to join the Group. The valid values are "all", "followers", or "admins". <b>Default:</b> "all".	Optional	
<b>UploadPolicy</b>	String	The policy that determines who can upload Content to the Group. The valid values are "all", "followers", "internal_followers", or "admins". <b>Default:</b> "all".	Optional	
<b>ModerationPolicy</b>	Boolean	The policy that determines whether Group administrators must approve member contributions to the Group, including blog posts, documents, photos, videos, feed posts, forum posts, and wiki pages. True if Group administrators must approve contributions; false if not. <b>Default:</b> "false".	Optional	
<b>GroupType</b>	String	The type of the Group. The valid values are "public_internal", "private_internal", or "private_external". <b>Default:</b> "public_internal".	Optional	

Property	Type	Description	Creata-ble	N
<b>AutoGroup</b>	Boolean	Whether the group is configured to be automatically populated with members according to preset criteria (based on Member Lists). See " <a href="#">Create and manage member lists</a> " and " <a href="#">Create an auto group</a> " in the <i>SAP Jam Collaboration Administrator Guide</i> for details. <b>Default:</b> "false".	Optional	
<b>DisableAtNotify</b>	Boolean	If set to true, members are not able to use @@notify (notify all) in conversations in the Group. <b>Default:</b> "false".	Optional	
<b>TermsOfUse</b>	String	The full text of the "Terms of Use" that members must agree to in order to access the Group. For members who have accepted the Terms of Use, or for Groups where no Terms of Use has been set, this property will have a null value. <b>Default:</b> (no content).	Optional	
<b>ContentsVisible</b>	Boolean	True if the Group's Content section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>QuestionsVisible</b>	Boolean	True if the Group's Questions (in Forums) is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>IdeasVisible</b>	Boolean	True if the Group's Ideas (in Forums) is set to be visible and available for use in to its Members. <b>Default:</b> "false".	Optional	
<b>DiscussionsVisible</b>	Boolean	True if the Group's Discussions (in Forums) is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>TasksVisible</b>	Boolean	True if the Group's Tasks feature is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>EventsVisible</b>	Boolean	True if the Group's Events feature is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>LinksVisible</b>	Boolean	True if the Group's Links section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>SubgroupsVisible</b>	Boolean	True if the Group's Subgroups section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional	
<b>RecommendationsVisible</b>	Boolean	True if the Recommendations widget is set to be visible and available in the Group for use by its Members. <b>Default:</b> "false".	Optional	

To summarize these requirements of the request to create a group in which you set all properties, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H "Content-Type: application/json" -H "Accept: application/json"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups" -d@C:/
payloads/payload.json
```

This next example for the JSON work flow shows the contents of the payload file:

```
{
  "Name": "Customer Request Discussion Group",
  "Description": "A group to track customer requests",
  "IsActive": false,
```



```

    "AutoSubscribe":true,
    "Announcement":"Please be sure to record all customer requests in this
group.",
    "OverviewAsLanding":true,
    "Participation":"full",
    "InvitePolicy":"followers",
    "UploadPolicy":"followers",
    "ModerationPolicy":true,
    "GroupType":"private_external",
    "AutoGroup":true,
    "DisableAtNotify":true,
    "TermsOfUse":"The restrictions of the 'Customer Information' section of your
NDA applies to this group.",
    "ContentsVisible":true,
    "QuestionsVisible":true,
    "IdeasVisible":true,
    "DiscussionsVisible":true,
    "TasksVisible":true,
    "EventsVisible":true,
    "LinksVisible":true,
    "SubgroupsVisible":true,
    "RecommendationsVisible":true
}

```

Alternatively, here is the curl command, if you choose to use the XML format:

```

curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups" -dC:/
payloads/payload.xml

```

This next example for the XML work flow shows the contents of the payload file:

```

<entry>
  <content>
    <properties>
      <Name>Customer Requests Decision Group</Name>
      <Description>A group to track customer requests</Description>
      <IsActive>false</IsActive>
      <AutoSubscribe>true</AutoSubscribe>
      <Announcement>Please be sure to record all customer requests in this
group.</Announcement>
      <OverviewAsLanding>true</OverviewAsLanding>
      <Participation>full</Participation>
      <InvitePolicy>followers</InvitePolicy>
      <UploadPolicy>followers</UploadPolicy>
      <ModerationPolicy>true</ModerationPolicy>
      <GroupType>private_external</GroupType>
      <AutoGroup>true</AutoGroup>
      <DisableAtNotify>true</DisableAtNotify>
      <TermsOfUse>The restrictions of the 'Customer Information' section
of your NDA applies to this group.</TermsOfUse>
      <ContentsVisible>true</ContentsVisible>
      <QuestionsVisible>true</QuestionsVisible>
      <IdeasVisible>true</IdeasVisible>
      <DiscussionsVisible>true</DiscussionsVisible>
      <TasksVisible>true</TasksVisible>
      <EventsVisible>true</EventsVisible>
      <LinksVisible>true</LinksVisible>
      <SubgroupsVisible>true</SubgroupsVisible>
      <RecommendationsVisible>true</RecommendationsVisible>
    </properties>
  </content>
</entry>

```

## Develop the ability to handle either the success or the error response

If your POST request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that creates a resource, such as this request, the response code is:

```
201 Created
```

- For any POST operation that creates a resource, you also receive a payload of the full information about the created resource.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": {
      "metadata": {
        "uri": "Groups('s7K9E4ohNFyL005kX1KJAh')",
        "type": "SAPJam.Group"
      },
      "Id": "s7K9E4ohNFyL005kX1KJAh",
      "Name": "Customer Requests Discussion Group",
      "Description": "A group to track customer requests",
      "IsActive": false,
      "AutoSubscribe": true,
      "Announcement": "Please be sure to record all customer requests in this group.",
      "OverviewAsLanding": true,
      "Participation": "expert",
      "InvitePolicy": "followers",
      "UploadPolicy": "followers",
      "ModerationPolicy": true,
      "GroupType": "private_external",
      "CreatedAt": "/Date(1413504563000)/",
      "LastModifiedAt": "/Date(1413504563000)/",
      "LastActivityAt": "/Date(1413504563000)/",
      "MembersCount": 1,
      "AutoGroup": false,
      "DisableAtNotify": false,
      "TermsOfUse": "The restrictions of the 'Customer Information' section of your NDA applies to this group.",
      "WebURL": "https://{jam#} [page 246].sapjam.com/groups/s7K9E4ohNFyL005kX1KJAh",
      "ContentsVisible": true,
      "QuestionsVisible": true,
      "IdeasVisible": true,
      "DiscussionsVisible": true,
      "TasksVisible": true,
      "EventsVisible": true,
      "LinksVisible": true,
      "SubgroupsVisible": true,
      "RecommendationsVisible": true,
      <!--[Navigation Properties have been removed from here.]-->
    }
  }
}
```

### i Note

In the above example, the many navigation URIs have been cut out to keep the code examples concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?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">
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('jHNQuNuVAV45kfySSRtD63')</id>
    <title type="text">Groups('jHNQuNuVAV45kfySSRtD63')</title>
    <updated>2016-06-22T00:06:10+00:00</updated>
    <author>
        <name/>
    </author>
    <link rel="edit" title="Groups('jHNQuNuVAV45kfySSRtD63') "
href="Groups('jHNQuNuVAV45kfySSRtD63')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Group" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
        <m:properties>
            <d:Id m:type="Edm.String">jHNQuNuVAV45kfySSRtD63</d:Id>
            <d:Name m:type="Edm.String">Customer Requests Decision Group</
d:Name>
            <d:Description m:type="Edm.String">A group to track customer
requests</d:Description>
            <d:IsActive m:type="Edm.Boolean">>false</d:IsActive>
            <d:AutoSubscribe m:type="Edm.Boolean">>true</d:AutoSubscribe>
            <d:Announcement m:type="Edm.String">Please be sure to record all
customer requests in this group.</d:Announcement>
            <d:OverviewAsLanding m:type="Edm.Boolean">>true</
d:OverviewAsLanding>
            <d:Participation m:type="Edm.String">full</d:Participation>
            <d:InvitePolicy m:type="Edm.String">followers</d:InvitePolicy>
            <d:UploadPolicy m:type="Edm.String">followers</d:UploadPolicy>
            <d:ModerationPolicy m:type="Edm.Boolean">>true</d:ModerationPolicy>
            <d:GroupType m:type="Edm.String">private_external</d:GroupType>
            <d:CreatedAt m:type="Edm.DateTimeOffset">2016-06-22T00:06:08Z</
d:CreatedAt>
            <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-06-22T00:06:08Z</d:LastModifiedAt>
            <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-06-22T00:06:08Z</d:LastActivityAt>
            <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
            <d:AutoGroup m:type="Edm.Boolean">>false</d:AutoGroup>
            <d:DisableAtNotify m:type="Edm.Boolean">>true</d:DisableAtNotify>
            <d:TermsOfUse m:type="Edm.String" m:null="true"/>
            <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/jHNQuNuVAV45kfySSRtD63</d:WebURL>
            <d:ContentsVisible m:type="Edm.Boolean">>true</d:ContentsVisible>
            <d:QuestionsVisible m:type="Edm.Boolean">>true</d:QuestionsVisible>
            <d:IdeasVisible m:type="Edm.Boolean">>true</d:IdeasVisible>
            <d:DiscussionsVisible m:type="Edm.Boolean">>true</
d:DiscussionsVisible>
            <d:TasksVisible m:type="Edm.Boolean">>true</d:TasksVisible>
            <d:EventsVisible m:type="Edm.Boolean">>true</d:EventsVisible>
            <d:LinksVisible m:type="Edm.Boolean">>true</d:LinksVisible>
            <d:SubgroupsVisible m:type="Edm.Boolean">>true</d:SubgroupsVisible>
            <d:RecommendationsVisible m:type="Edm.Boolean">>true</
d:RecommendationsVisible>
            <d:HasOverview m:type="Edm.Boolean">>false</d:HasOverview>
            <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
            <d:IsAdmin m:type="Edm.Boolean">>true</d:IsAdmin>
        </m:properties>
    </content>
</entry>

```

## i Note

In the above example, the many navigation URIs have been cut out to keep the code examples concise.

If your POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About POST calls

The unique aspects of using POST calls are:

- You do not set the 'Id' of a resource as a URL parameter.
- You do not use query parameters, except for uploading content, which is shown in the later lesson, [Uploading and downloading files \[page 359\]](#).
- You must include a payload that specifies the properties of the resource that you want to create.
- The success code for a call that creates a resource is `201 Created`.
- For a call that creates a resource, a payload listing all properties of the created resource is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to create a group setting most or all available options \[page 266\]](#)?
- Do you know what the [possible responses \[page 270\]](#) are so that you can develop the ability to handle these responses.
- Do you understand the [unique aspects of a POST call \[page 272\]](#) so that you can easily perform other POST operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Copy an existing group \[page 273\]](#).

### 8.1.2.3.3 Copy an existing group

This page of "The Groups and related API" shows you how to use an alternate way to create a group by using the [POST] /Group\_Copy endpoint. This method copies an existing group to create a new group. This is the first explanation of how to use a "service operation".

#### Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to copy an existing Group \[page 273\]](#).
- Know what the [possible responses \[page 274\]](#) are to your request so that you can develop the ability to handle these responses.
- Understand the [unique aspects of a POST service operation call \[page 276\]](#) so that you can easily perform other POST service operations.

#### Develop the service operation POST request to copy a group


If you have a Group that you would like to use as the basis for another Group, you can use the [POST] /Group\_Copy API call to create a duplicate of the original Group. For this operation, you identify the Group that you want to copy, and you provide the name of new group.

This section shows the information that must be passed in a POST /Group\_Copy request to create a group by copying an existing group.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "POST" request.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_Copy?  
Id='{Id}'&NewName='{Name}'
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "?Id='{Id}' ", which sets the unique identifier of the group resource that you want to copy (replace {Id} with that identifier).
- "&NewName='{Name}' ", which is the new name that the currently logged-in user wants to set for the new group created by the copy operation (replace {Name} with that identifier). Note that any spaces or special characters in the new group name must be encoded as described in [RFC2396: Uniform Resource Identifiers \(URI\): Generic Syntax](#) ; for example, spaces must be encoded as "%20".

To summarize these requirements of the request to create a group in which you set all properties, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
      -H "Content-Type: application/json" -H "Accept: application/json"
      -X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_Copy?
      Id='QOXAzcxqzVVbQ6tQjUqCKu'&NewName='Customer%20Service%20Department%20Group'"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer As3UvIaYEvDXoeREtmSz3qeCpnNvrrHZhVMswcBV'
      -H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
      -X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_Copy?
      Id='BrxJdwnaPLf8AUlP6L08TY'&NewName='Name%20of%20New%20Group'"
```

## Develop the ability to handle either the success or the error response

If your POST request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that creates a resource, such as this request, the response code is:

```
201 Created
```

- For any POST operation that creates a resource, you also receive a payload of the full information about the created resource.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "Groups('JuzRMazLNGQz2295wGWqAu')",
        "type": "SAPJam.Group"
      },
      "Id": "JuzRMazLNGQz2295wGWqAu",
      "Name": "Customer Service Department Group",
      "Description": "",
      "IsActive": true,
      "AutoSubscribe": false,
      "Announcement": null,
      "OverviewAsLanding": true,
      "Participation": "full",
      "InvitePolicy": "all",
      "UploadPolicy": "all",
      "ModerationPolicy": false,
      "GroupType": "public_internal",
      "CreatedAt": "/Date(1466717438000)/",
      "LastModifiedAt": "/Date(1466717439000)/",
      "LastActivityAt": "/Date(1466717438000)/",
      "MembersCount": 1,
      "AutoGroup": false,
      "DisableAtNotify": false,
      "TermsOfUse": null,
      "WebURL": "https://{jam#} [page 246].sapjam.com/groups/JuzRMazLNGQz2295wGWqAu",
      "ContentsVisible": true,
      "QuestionsVisible": true,
      "IdeasVisible": true,
      "DiscussionsVisible": true,
      "TasksVisible": true
    }
  }
}
```

```

    "EventsVisible": true,
    "LinksVisible": true,
    "SubgroupsVisible": true,
    "RecommendationsVisible": true,
    "HasOverview": false,
    "TaskPolicy": "readonly",
    "IsAdmin": true,
    /* [Navigation Properties have been removed from here.] */
  }}}

```

## i Note

In the above example, the many navigation URLs have been cut out to keep the code examples concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/odata/
Groups('N330DxkPkfk3SDX4zdozbI')</id>
  <title type="text">Groups('N330DxkPkfk3SDX4zdozbI')</title>
  <updated>2016-06-23T21:52:23+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Groups('N330DxkPkfk3SDX4zdozbI') "
href="Groups('N330DxkPkfk3SDX4zdozbI')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Group" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">N330DxkPkfk3SDX4zdozbI</d:Id>
      <d:Name m:type="Edm.String">Customer Support Department Group</
d:Name>
      <d:Description m:type="Edm.String"/>
      <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
      <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
      <d:Announcement m:type="Edm.String" m:null="true"/>
      <d:OverviewAsLanding m:type="Edm.Boolean">true</
d:OverviewAsLanding>
      <d:Participation m:type="Edm.String">full</d:Participation>
      <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
      <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
      <d:ModerationPolicy m:type="Edm.Boolean">false</
d:ModerationPolicy>
      <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2016-06-23T21:52:22Z</
d:CreatedAt>
      <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-06-23T21:52:23Z</d:LastModifiedAt>
      <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-06-23T21:52:22Z</d:LastActivityAt>
      <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
      <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
      <d:DisableAtNotify m:type="Edm.Boolean">false</d:DisableAtNotify>
      <d:TermsOfUse m:type="Edm.String" m:null="true"/>
      <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/N330DxkPkfk3SDX4zdozbI</d:WebURL>
      <d:ContentsVisible m:type="Edm.Boolean">true</d:ContentsVisible>
      <d:QuestionsVisible m:type="Edm.Boolean">true</d:QuestionsVisible>
      <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>

```

```

        <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">true</d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">true</d:IsAdmin>
    </m:properties>
</content>
</entry>

```

### Note

In the above example, the many navigation URIs have been cut out to keep the code examples concise.

If your service operation POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About service operation calls

The unique aspects of using a service operation call that creates a resource are:

- You must set any query parameters that are required for that particular service operation.
- The success code for a call that creates a resource is 201 `Created`.
- For a call that creates a resource, a payload listing all properties of the created resource is returned.



## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to formulate a request to [copy an existing Group \[page 273\]](#)?
- Do you know what the [possible responses \[page 274\]](#) are to your request so that you can develop the ability to handle these responses?
- Do you understand the [unique aspects of a POST service operation call \[page 276\]](#) so that you can perform other POST service operation calls?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a group's information \[page 277\]](#).

### 8.1.2.3.4 Retrieve a group's information

This page of "The Groups and related API" shows you how to retrieve a single existing Group's information using the `[GET] /Groups('{id}')` endpoint. This is the first explanation of how to use a GET operation.

## Learning Objectives

The learning objectives for this page are:

- Know how to formulate a request to [copy and existing Group \[page 277\]](#).
- Know what the [possible responses \[page 278\]](#) are to your request so that you can develop the ability to handle these responses.
- Understand the [unique aspects of a GET call \[page 280\]](#) so that you can easily perform other GET operations.

## Develop the GET request to retrieve a group's information

This section shows the information that is required to retrieve a group's information by using the `[GET] /Group('{Id}')` endpoint.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" request.

4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Groups('RsOpnZp4b12KRc5KR8vY4Q')
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Group resource whose information you want to retrieve.

To summarize these requirements of the request to retrieve a group's information, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed  
from here.] */"  
-H "Content-Type: application/json" -H "Accept: application/json"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Groups('RsOpnZp4b12KRc5KR8vY4Q')"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed  
from here.]-->"  
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Groups('RsOpnZp4b12KRc5KR8vY4Q')"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a successful GET request, the response code is:

```
200 OK
```

- You also receive a payload of the full information of the retrieved resource.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": { "results": {  
  "__metadata": {  
    "uri": "Groups('RsOpnZp4b12KRc5KR8vY4Q')",  
    "type": "SAPJam.Group"  
  },  
  "Id": "RsOpnZp4b12KRc5KR8vY4Q",  
  "Name": "Customer Requests Discussion Group",  
  "Description": "",  
  "IsActive": true,  
  "AutoSubscribe": false,  
  "Announcement": null,  
  "OverviewAsLanding": true,  
  "Participation": "full",  
  "InvitePolicy": "all",  
  "UploadPolicy": "all",  
  "ModerationPolicy": false,  
  "GroupType": "public internal",  
  "CreatedAt": "/Date(1466444431000)/",  
  "LastModifiedAt": "/Date(1466444431000)/",  
  "LastActivityAt": "/Date(1466444432000)/",  
  "MembersCount": 1,  
  "Members": []  
}}
```

```

    "AutoGroup": false,
    "DisableAtNotify": false,
    "TermsOfUse": null,
    "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
RsOpnZp4b12KRc5KR8vY4Q",
    "ContentsVisible": true,
    "QuestionsVisible": true,
    "IdeasVisible": true,
    "DiscussionsVisible": true,
    "TasksVisible": true,
    "EventsVisible": true,
    "LinksVisible": true,
    "SubgroupsVisible": true,
    "RecommendationsVisible": true,
    "HasOverview": false,
    "TaskPolicy": "readonly",
    "IsAdmin": true,
    /* [Navigation Properties have been removed from here.] */
  }}}

```

## Note

In the above example, the many navigation URIs have been cut out to keep the code examples concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('RsOpnZp4b12KRc5KR8vY4Q')</id>
  <title type="text">Groups('RsOpnZp4b12KRc5KR8vY4Q')</title>
  <updated>2016-07-06T19:27:10+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Groups('RsOpnZp4b12KRc5KR8vY4Q') "
href="Groups('RsOpnZp4b12KRc5KR8vY4Q')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Group" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">RsOpnZp4b12KRc5KR8vY4Q</d:Id>
      <d:Name m:type="Edm.String">Customer Requests Discussion Group</
d:Name>
      <d:Description m:type="Edm.String"/>
      <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
      <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
      <d:Announcement m:type="Edm.String" m:null="true"/>
      <d:OverviewAsLanding m:type="Edm.Boolean">true</
d:OverviewAsLanding>
      <d:Participation m:type="Edm.String">full</d:Participation>
      <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
      <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
      <d:ModerationPolicy m:type="Edm.Boolean">false</
d:ModerationPolicy>
      <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</
d:CreatedAt>
      <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-06-20T17:40:31Z</d:LastModifiedAt>
      <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-06-20T17:40:32Z</d:LastActivityAt>

```

```

    <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
    <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
    <d:DisableAtNotify m:type="Edm.Boolean">false</d:DisableAtNotify>
    <d:TermsOfUse m:type="Edm.String" m:null="true"/>
    <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/RsOpnZp4b12KRc5KR8vY4Q</d:WebURL>
    <d:ContentsVisible m:type="Edm.Boolean">true</d:ContentsVisible>
    <d:QuestionsVisible m:type="Edm.Boolean">true</d:QuestionsVisible>
    <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>
    <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
    <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
    <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
    <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
    <d:SubgroupsVisible m:type="Edm.Boolean">true</d:SubgroupsVisible>
    <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
    <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
    <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
    <d:IsAdmin m:type="Edm.Boolean">true</d:IsAdmin>
  </m:properties>
</content>
</entry>

```

### Note

In the above example, the many navigation URIs have been cut out to keep the code examples concise.

If your GET request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About GET calls

The unique aspects of using GET calls are:

- If you want to retrieve the information about a specific resource, you must set the 'Id' of the resource as a URL parameter.

- You may use query parameters to manage the resources that are returned, although that is shown in the later lesson, [OData Query Parameters \[page 323\]](#).
- You do not include a payload in the request.
- The success code for a call that retrieves the information on a specific resource is 200 OK.
- For a successful GET call, a payload listing all properties of the requested resource is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to formulate a request to [retrieve a Group's information \[page 277\]](#)?
- Do you know what the [possible responses \[page 278\]](#) are to your request so that you can develop the ability to handle these responses?
- Do you understand the [unique aspects of a GET call \[page 280\]](#) so that you can easily perform other GET operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [About group templates \[page 281\]](#).

### 8.1.2.3.5 About group templates

This section of "The Groups and related API" lesson introduces you to the use of the GroupTemplate, SystemGroupTemplate, and CustomGroupTemplate entities. The GroupTemplates entity is a convenient mechanism for accessing both SystemGroupTemplates and CustomGroupTemplates. SystemGroupTemplates are preconfigured and packaged with SAP Jam. CustomGroupTemplates are templates that you or others in your organization create.

The information that is saved in any GroupTemplate includes:

- Forums (Discussions, Ideas, Questions)
- blogs

But does not include:

- Content (except for the blogs)
- Any Folders structure
- FeedEntries

This section shows you how to:

- [Save a group as a CustomGroupTemplate \[page 282\]](#).  
You can save a group that is configured for a particular use as a CustomGroupTemplate by using the [POST] /Group\_SaveAsTemplate Service Operation.
- [Update a CustomGroupTemplate \[page 286\]](#).  
Of all the GroupTemplate entities, only the CustomGroupTemplate can be updated. This is done by using the [PATCH] /CustomGroupTemplates('{id}') endpoint. This allows you to update the template title and description, as well as to set whether the CustomGroupTemplate is enabled for use as a template for new groups.

- Know how to [Retrieve a feed of available GroupTemplates \[page 289\]](#).  
You can retrieve a "feed" or "collection" of all of the GroupTemplates that are available to the current user by using the [GET] /GroupTemplates endpoint. Any GET operation that does not specify a particular resource by setting its unique ID in the URL (and that is not a service operation) will return a collection of resources. These are referred to as collection-valued GET requests.
- [Create a group from a GroupTemplate \[page 294\]](#).  
The purpose of GroupTemplates is to create a new Group based on the selected template. This is also done with the same [POST] /Groups endpoint, but in this operation you specify the name of the new Group and provide the URI of the GroupTemplate that you want to base the new Group on.

### 8.1.2.3.5.1 Save a group as a CustomGroupTemplate

This page of "The Groups and related API" shows you how to save a group as a CustomGroupTemplate by using the [POST] /Group\_SaveAsTemplate service operation so that you can later create similar-purpose groups. This is the second explanation of how to use a "service operation".

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to save a Group as a GroupTemplate \[page 282\]](#).
- Know what the [possible responses \[page 283\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a POST service operation \[page 285\]](#) so that you can easily perform other POST service operations.

## Develop the request to save a group as a CustomGroupTemplate

This section shows the information that is required to form an effective request to save a group as CustomGroupTemplate by using the [POST] /Group\_SaveAsTemplate service operation endpoint.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "POST" request.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_SaveAsTemplate
?Id='{Id}'&Title='{Title}'
&Description='{Description}'&Enabled={true|false}"
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- Set the required parameters of this service operation request as query options. The required parameters are:
  - Parameter: Id=' {Id} ', where {Id} is the unique identifier of the Group that you want to save as a CustomGroupTemplate.
  - Parameter: Title=' {Title} ', where {Title} is the title or name that you want to give to the created CustomGroupTemplate.
  - Parameter: Description=' {Description} ', where {Description} is the description that you want to give to the created CustomGroupTemplate.
  - Parameter: Enabled={true|false}, where {true|false} is either "true" or "false", which indicates whether you want the created CustomGroupTemplate to be visible and available to Members to create new Groups from.

### Note

Any spaces or special characters in the "Title" or the "Description" must be encoded as described in [RFC2396: Uniform Resource Identifiers \(URI\): Generic Syntax](#); for example, spaces must be encoded as "%20".

To summarize these requirements of the request to create a group, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */" \
  -H "Content-Type: application/json" -H "Accept: application/json" \
  -X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_SaveAsTemplate?Id='RsOpnZp4b12KRc5KR8vY4Q'&Title='Customer%20Request%20Tracking%20Template'&Description='Use%20for%20tracking%20all%20customer%20feature%20requests'&Enabled=true"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: <!--[Navigation Properties have been removed from here.]-->" \
  -H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml" \
  -X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Group_SaveAsTemplate?Id='RsOpnZp4b12KRc5KR8vY4Q'&Title='Consumer%20Request%20Tracking%20Template'&Description='Use%20for%20tracking%20all%20customer%20feature%20requests'&Enabled=true"
```

## Develop the ability to handle either the success or the error response

If your POST service operation request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that creates a resource, such as this request, the response code is:

```
201 Created
```

- You also receive a payload of the full information of the created resource.  
If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": {
      "metadata": {
        "uri": "CustomGroupTemplates('RkrRW2Vv6OB0okketYgp9g')",
        "type": "SAPJam.CustomGroupTemplate"
      },
      "Id": "RkrRW2Vv6OB0okketYgp9g",
      "Title": "Customer Request Tracking Template",
      "Description": "Use for tracking all customer feature requests",
      "Enabled": true,
      "Locale": "en"
    }
  }
}
```

### Note

That there are no navigation URIs for CustomGroupTemplates.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?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">
  <id>https://[jam#] [page 246].sapjam.com/api/v1/OData/CustomGroupTemplates('DmaWGQWTidRq7OnUMovEIY')</id>
  <title type="text">CustomGroupTemplates('DmaWGQWTidRq7OnUMovEIY')</title>
  <updated>2016-07-06T22:31:09+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="CustomGroupTemplates('DmaWGQWTidRq7OnUMovEIY') " href="CustomGroupTemplates('DmaWGQWTidRq7OnUMovEIY') ">
    <category term="SAPJam.CustomGroupTemplate" scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">DmaWGQWTidRq7OnUMovEIY</d:Id>
        <d:Title m:type="Edm.String">Customer Request Tracking Template</d:Title>
        <d:Description m:type="Edm.String">Use for tracking all customer feature requests</d:Description>
        <d:Enabled m:type="Edm.Boolean">true</d:Enabled>
        <d:Locale m:type="Edm.String">en</d:Locale>
      </m:properties>
    </content>
  </entry>
```

### Note

That there are no navigation URIs for CustomGroupTemplates.

If your POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).



- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a ContentType that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About service operation calls

The unique aspects of using a service operation call that creates a resource are:

- You must set any query parameters that are required for that particular service operation.
- The success code for a call that creates a resource is 201 `Created`.
- For a call that creates a resource, a payload listing all properties of the created resource is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to save a Group as a GroupTemplate \[page 282\]](#)?
- Do you know what the [possible responses \[page 283\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a POST service operation \[page 285\]](#) so that you can easily perform other POST service operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Update a CustomGroupTemplate \[page 286\]](#).

## 8.1.2.3.5.2 Update a CustomGroupTemplate

This page of "The Groups and related API" shows you how to update a CustomGroupTemplate by using the [PATCH] /CustomGroupTemplates('{id}') endpoint. This is the first explanation of how to use a PATCH operation.

### Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the PATCH request to update a CustomGroupTemplate \[page 286\]](#).
- Know what the [possible responses \[page 287\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a PATCH call \[page 288\]](#) so that you can easily perform other PATCH operations.

### Develop the PATCH request to update a CustomGroupTemplate

This section shows the minimal information that is required to update a CustomGroupTemplate by using the [PATCH] /CustomGroupTemplates('{id}') endpoint.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "PATCH" request.
4. Set the URL for your request, which will be:

```
https://{jam#} \[page 246\].sapjam.com/api/v1/OData/CustomGroupTemplates('{id}')
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
  - "{Id}", which is the unique identifier of the CustomGroupTemplate resource that you want to update.
5. Set the properties that you want to update in a payload file. You can update the following properties:
    - **Title:** The name of the CustomGroupTemplate.
    - **Description:** The description of the CustomGroupTemplate.
    - **Enabled:** Whether the CustomGroupTemplate will be set to be visible and available for users to create a new group.

To summarize these requirements of the request to create a group, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */"
```

```
-H "Content-Type: application/json" -H "Accept: application/json"
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/
CustomGroupTemplates('RsOpnZp4b12KRc5KR8vY4Q') "
-d@C:/payloads/payload.json
```

The contents of the `payload.json` file will look something like this, depending on what properties you want to set:

```
{
  "Title": "Customer Request Tracking Template",
  "Description": "Use for tracking all customer feature requests",
  "Enabled": false
}
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/
CustomGroupTemplates('RsOpnZp4b12KRc5KR8vY4Q') "
-d@C:/payloads/payload.xml
```

The contents of the `payload.xml` file will look something like this, depending on what properties you want to set:

```
<entry>
  <content>
    <properties>
      <Title>Customer Request Tracking Template</Title>
      <Description>Use for tracking all customer feature requests</
Description>
      <Enabled>false</Enabled>
    </properties>
  </content>
</entry>
```

## Develop the ability to handle either the success or the error response

If your PATCH request is successful, the following information is returned:

- You receive an HTTP success code. For a PATCH request, such as this request, the response code is:

```
204 No Content
```

### Note

You do not receive a payload in response to any PATCH request.

If your POST request is not successful, one of the following error codes is returned:

- 400 Bad Request:** The server could not understand the request due to malformed syntax.
- 403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- 404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).

- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a ContentType that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About PATCH calls

The unique aspects of using a service operation call that creates a resource are:

- To update a specific resource, you must set the 'Id' of that resource as a URL parameter.
- You do not use query parameters, except for uploading content, as is shown in the later lesson, [Uploading and downloading files \[page 359\]](#).
- You must include a payload that specifies the properties of the resource that you want to create.
- The success code for a PATCH request is always 204 No Content.
- No payload is returned to a PATCH request.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the PATCH request to update a CustomGroupTemplate \[page 286\]](#)?
- Do you know what the [possible responses \[page 287\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a PATCH call \[page 288\]](#) so that you can perform other similar operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a feed of available GroupTemplates \[page 289\]](#).

## 8.1.2.3.5.3 Retrieve a feed of available GroupTemplates

This page of "The Groups and related API" lesson shows you how to retrieve a feed of the available GroupTemplates by using the collection-valued API call, `[GET] /GroupTemplates`. This is the second explanation of how to use a GET operation.

### Learning Objectives

The learning objectives for this page are:

- Know what information is required to form the [GET /GroupTemplates request \[page 289\]](#).
- Know what the [possible responses \[page 290\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a collection-valued GET call \[page 294\]](#) so that you can easily perform other collection-valued GET operations.

### Develop the call to retrieve a feed of available GroupTemplates

This section shows the information that is required to retrieve a feed of information on the GroupTemplates that are available to the current user by using the `GET /GroupTemplates` endpoint.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/GroupTemplates
```

Where the only variable URL parameter for this call is:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".

To summarize these requirements of the request to create a group, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */" \
  -H "Content-Type: application/json" -H "Accept: application/json" \
  "https://{jam#} [page 246].sapjam.com/api/v1/OData/GroupTemplates"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->" \
  -H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
```

```
"https://{jam#} [page 246].sapjam.com/api/v1/OData/GroupTemplates"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- You also receive a payload of information on up to 20 GroupTemplates entries.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": {
  "results": [
    {
      "__metadata": {
        "uri":
"GroupTemplates(Id='MydCkBCBzuEss4iolUJApS',GroupTemplateType='system')",
        "type": "SAPJam.GroupTemplate"
      },
      "Id": "MydCkBCBzuEss4iolUJApS",
      "GroupTemplateType": "system",
      "Title": "Help & Support Template",
      "Description": "An internal-facing Q&A community designed for
members and experts to help each other on a particular topic of interest.",
      "Enabled": true
    },
    {
      "__metadata": {
        "uri":
"GroupTemplates(Id='tdyoy6Z4tqTJJX6hIjtPcA',GroupTemplateType='system')",
        "type": "SAPJam.GroupTemplate"
      },
      "Id": "tdyoy6Z4tqTJJX6hIjtPcA",
      "GroupTemplateType": "system",
      "Title": "Team Collaboration Template",
      "Description": "A group that helps members stay organized, share
resources, and create a virtual presence for their existing team.",
      "Enabled": true
    },
    /* [Eleven "system" GroupTemplates entries have been cut from here.]
  */
    {
      "__metadata": {
        "uri":
"GroupTemplates(Id='uA1MXCSbx6d3bfrD6v4X24',GroupTemplateType='custom')",
        "type": "SAPJam.GroupTemplate"
      },
      "Id": "uA1MXCSbx6d3bfrD6v4X24",
      "GroupTemplateType": "custom",
      "Title": "ACE Corp HR Mentoring template",
      "Description": "Group for Ask-the-Mentor program at ACE",
      "Enabled": true
    },
    {
      "__metadata": {
        "uri":
"GroupTemplates(Id='xpQ3OGXqOcsDRn9fc9ulZ2',GroupTemplateType='custom')",
        "type": "SAPJam.GroupTemplate"
      },
      "Id": "xpQ3OGXqOcsDRn9fc9ulZ2",
```

```

        "GroupTemplateType": "custom",
        "Title": "Planning and Implementation Custom template",
        "Description": "Planning and Implementation Custom template for
Talent LoB",
        "Enabled": true
    },
    /* [Five "custom" GroupTemplates entries have been cut from here.] */
  ],
  "__next": "GroupTemplates?%24skiptoken=lyhyr"
}}

```

## i Note

That there are no navigation URIs for GroupTemplates. Also, note that 16 of the records for GroupTemplates have been removed to keep this code example brief.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData"
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>GroupTemplates</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/GroupTemplates</id>
  <link rel="self" title="GroupTemplates"
href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates"/>
  <link rel="next" title="GroupTemplates?%24skiptoken=lyhyr"
href="GroupTemplates?%24skiptoken=lyhyr"/>
  <updated>2016-07-11T23:59:05+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates(Id='MydCkBCBzuEss4iolUJApS',GroupTemplateType='system')</id>
    <title type="text"

>GroupTemplates(Id='MydCkBCBzuEss4iolUJApS',GroupTemplateType='system')</
title>
    <updated>2016-07-11T23:59:05+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit"

title="GroupTemplates(Id='MydCkBCBzuEss4iolUJApS',GroupTemplateType='system')
href="GroupTemplates(Id='MydCkBCBzuEss4iolUJApS',GroupTemplateType='system')"/
>
    <category term="SAPJam.GroupTemplate"
scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">MydCkBCBzuEss4iolUJApS</d:Id>
        <d:GroupTemplateType m:type="Edm.String">system</
d:GroupTemplateType>
        <d:Title m:type="Edm.String">Help & Support Template</d:Title>
        <d:Description m:type="Edm.String">An internal-facing Q&A
community designed for
members and experts to help each other on a particular
topic of
interest.</d:Description>
        <d:Enabled m:type="Edm.Boolean">true</d:Enabled>
      </m:properties>
    </content>
  </entry>
</feed>

```

```

    </entry>
    <entry>
      <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates(Id='tdyoy6Z4tqTJJX6hIjtPcA',GroupTemplateType='system')</id>
      <title type="text">

>GroupTemplates(Id='tdyoy6Z4tqTJJX6hIjtPcA',GroupTemplateType='system')</
title>
      <updated>2016-07-11T23:59:05+00:00</updated>
      <author>
        <name/>
      </author>
      <link rel="edit"

title="GroupTemplates(Id='tdyoy6Z4tqTJJX6hIjtPcA',GroupTemplateType='system')>
href="GroupTemplates(Id='tdyoy6Z4tqTJJX6hIjtPcA',GroupTemplateType='system')"/
>
      <category term="SAPJam.GroupTemplate"
        scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
      <content type="application/atom+xml">
        <m:properties>
          <d:Id m:type="Edm.String">tdyoy6Z4tqTJJX6hIjtPcA</d:Id>
          <d:GroupTemplateType m:type="Edm.String">system</
d:GroupTemplateType>
          <d:Title m:type="Edm.String">Team Collaboration Template</
d:Title>
          <d:Description m:type="Edm.String">A group that helps members
stay organized, share
            resources, and create a virtual presence for their
existing
              team.</d:Description>
          <d:Enabled m:type="Edm.Boolean">true</d:Enabled>
        </m:properties>
      </content>
    </entry>
    <!-- [Eleven "system" GroupTemplates entries have been cut from here.] -->
    <entry>
      <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates(Id='uA1MXCSbx6d3bfrD6v4X24',GroupTemplateType='custom')</id>
      <title type="text">

>GroupTemplates(Id='uA1MXCSbx6d3bfrD6v4X24',GroupTemplateType='custom')</
title>
      <updated>2016-07-11T23:59:05+00:00</updated>
      <author>
        <name/>
      </author>
      <link rel="edit"

title="GroupTemplates(Id='uA1MXCSbx6d3bfrD6v4X24',GroupTemplateType='custom')>
href="GroupTemplates(Id='uA1MXCSbx6d3bfrD6v4X24',GroupTemplateType='custom')"/
>
      <category term="SAPJam.GroupTemplate"
        scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
      <content type="application/atom+xml">
        <m:properties>
          <d:Id m:type="Edm.String">uA1MXCSbx6d3bfrD6v4X24</d:Id>
          <d:GroupTemplateType m:type="Edm.String">custom</
d:GroupTemplateType>
          <d:Title m:type="Edm.String">ACE Corp HR Mentoring template</
d:Title>
          <d:Description m:type="Edm.String">Group for Ask-the-Mentor
program at
            ACE</d:Description>

```



```

        <d:Enabled m:type="Edm.Boolean">true</d:Enabled>
      </m:properties>
    </content>
  </entry>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates(Id='xpQ3OGXqOcsDRn9fc9ulZ2',GroupTemplateType='custom')</id>
    <title type="text"

>GroupTemplates(Id='xpQ3OGXqOcsDRn9fc9ulZ2',GroupTemplateType='custom')</
title>
    <updated>2016-07-11T23:59:05+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit"

title="GroupTemplates(Id='xpQ3OGXqOcsDRn9fc9ulZ2',GroupTemplateType='custom') "
href="GroupTemplates(Id='xpQ3OGXqOcsDRn9fc9ulZ2',GroupTemplateType='custom') "/
>
    <category term="SAPJam.GroupTemplate"
scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">xpQ3OGXqOcsDRn9fc9ulZ2</d:Id>
        <d:GroupTemplateType m:type="Edm.String">custom</
d:GroupTemplateType>
        <d:Title m:type="Edm.String">Planning and Implementation
Custom template</d:Title>
        <d:Description m:type="Edm.String">Planning and
Implementation Custom template for
Talent LoB</d:Description>
        <d:Enabled m:type="Edm.Boolean">true</d:Enabled>
      </m:properties>
    </content>
  </entry>
  <!-- [Eleven "system" GroupTemplates entries have been cut from here.] -->
</feed>

```

### i Note

That there are no navigation URIs for GroupTemplates. Also, note that 16 of the records for GroupTemplates have been removed to keep this code example brief.

If your POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a ContentType that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.

- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About collection-valued GET calls

The unique aspects of using [collection-valued \[page 243\]](#) GET calls are:

- If you want to retrieve the information about all resource of the specified entity type that are available to the currently logged-in user, you set the resource type as usual, but you must not set the 'Id' of a specific resource as a URL parameter.
- You may use query parameters to manage the resources that are returned, although that is shown in the later lesson, [OData Query Parameters \[page 323\]](#).
- As for all GET calls, you do not include a payload in the request.
- The success code for a call that returns a feed of resources is the as for all successful GET calls, 200 OK.
- For a successful collection-valued GET call, a payload listing multiple entries of all properties of the requested resource type is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form the [GET /GroupTemplates request \[page 289\]](#)?
- Do you know what the [possible responses \[page 290\]](#) are so that you can develop the ability to handle them?
- Do you understand the information returned to [collection-valued GET calls \[page 294\]](#)?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Create a group from a GroupTemplate \[page 294\]](#).

## 8.1.2.3.5.4 Create a group from a GroupTemplate

This page of "The Groups and related API" lesson shows you how to use a GroupTemplate to create a new Group. This is done with the same `[POST] /Groups` endpoint that you created Groups with in earlier pages of this lesson, but in this operation you specify the name of the new Group and provide the URI of the GroupTemplate that you want to base the new Group on. This is the third explanation of how to use a POST operation.

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form a [POST request for creating a group from a GroupTemplate \[page 295\]](#).
- Know what the [possible responses \[page 296\]](#) are to this request so that you can develop the ability to handle them.
- Understand the [unique aspects of a POST call \[page 298\]](#) so that you can easily perform other POST operations.

## Develop the POST request to create a Group from a GroupTemplate

This section shows the information that is required to create a group from a GroupTemplate by using the [POST] /Groups API call.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "POST" request.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups
```

Where the only variable URL parameter for this call is:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
5. In a payload file, set the name of the new group that you want to create in the form "Name": "New group from the 'Help and Support' template", where "New group from the 'Help and Support' template" should be set to whatever name the user wants to give to the new group. Also, specify the GroupTemplate that you want to use as the template for your new group.

To summarize these requirements of the request to create a group, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */"
-H "Content-Type: application/json" -H "Accept: application/json"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"
-d@C:/payloads/payload.json
```

As described, you provide the name of the new Group and the URI of the GroupTemplate in the request, or rather in the payload file specified in the request call.

The contents of the payload.json file is shown below:

```
{
  "Name": "New group from the 'Help and Support' template (using JSON format)",
```

```

    "Template":
    {
        "__metadata": {"uri": "https://{jam#} [page 246].sapjam.com/api/v1/
OData/
GroupTemplates (Id='MydCkBCBzuEss4iolUJApS', GroupTemplateType='system')"}
    }
}

```

Alternatively, here is the curl command, if you choose to use the XML format:

```

curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"
-d@C:/payloads/payload.xml

```

As described, you provide the name of the new Group and the URI of the GroupTemplate in the request, or rather in the payload file specified in the request call.

The contents of the `payload.xml` file is shown below:

```

<entry>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Template" href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
GroupTemplates (Id='MydCkBCBzuEss4iolUJApS', GroupTemplateType='system')"/>
  <content>
    <properties>
      <Name>New groups from the 'Help and Support' template (using XML
format)</Name>
    </properties>
  </content>
</entry>

```

## Develop the ability to handle either the success or the error response

If your POST request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that creates a resource, such as this request, the response code is:

```
201 Created
```

- You also receive a payload of the full information of the created resource. If you set the Accept header to "application/json", the returned payload will look like this:

```

{"d": {"results": {
  "__metadata": {
    "uri": "Groups('wgz6kUSWDIH6T4kjskeqHL')",
    "type": "SAPJam.Group"
  },
  "Id": "wgz6kUSWDIH6T4kjskeqHL",
  "Name": "New group from the 'Help and Support' template (using JSON
format)",
  "Description": "",
  "IsActive": true,
  "AutoSubscribe": false,
  "Announcement": null,
  "OverviewAsLanding": true,
  "Participation": "full",

```

```

    "InvitePolicy": "all",
    "UploadPolicy": "all",
    "ModerationPolicy": false,
    "GroupType": "public_internal",
    "CreatedAt": "/Date(1468282607000)/",
    "LastModifiedAt": "/Date(1468282607000)/",
    "LastActivityAt": "/Date(1468282607000)/",
    "MembersCount": 1,
    "AutoGroup": false,
    "DisableAtNotify": false,
    "TermsOfUse": null,
    "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
wgz6kUSWDIH6T4kjskeqHL",
    "ContentsVisible": true,
    "QuestionsVisible": true,
    "IdeasVisible": true,
    "DiscussionsVisible": true,
    "TasksVisible": true,
    "EventsVisible": true,
    "LinksVisible": true,
    "SubgroupsVisible": true,
    "RecommendationsVisible": true,
    "HasOverview": true,
    "TaskPolicy": "readonly",
    "IsAdmin": true,
    /* [Navigation Properties have been removed from here.] */
  }}}

```

## i Note

The the many navigation URIs for the created group are removed from the response example above to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('sbFeb19ZmYWakqXtHROdiq')</id>
  <title type="text">Groups('sbFeb19ZmYWakqXtHROdiq')</title>
  <updated>2016-07-12T00:35:10+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Groups('sbFeb19ZmYWakqXtHROdiq') "
href="Groups('sbFeb19ZmYWakqXtHROdiq')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Group" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">sbFeb19ZmYWakqXtHROdiq</d:Id>
      <d:Name m:type="Edm.String">New groups from the \"Help and Support
\" template (using XML
      format)</d:Name>
      <d:Description m:type="Edm.String"/>
      <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
      <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
      <d:Announcement m:type="Edm.String" m:null="true"/>
      <d:OverviewAsLanding m:type="Edm.Boolean">true</
d:OverviewAsLanding>
      <d:Participation m:type="Edm.String">full</d:Participation>
      <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
    </m:properties>
  </content>
</entry>

```

```

        <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
        <d:ModerationPolicy m:type="Edm.Boolean">>false</
d:ModerationPolicy>
        <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
        <d:CreatedAt m:type="Edm.DateTimeOffset">2016-07-12T00:35:05Z</
d:CreatedAt>
        <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-07-12T00:35:05Z</d:LastModifiedAt>
        <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-07-12T00:35:05Z</d:LastActivityAt>
        <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
        <d:AutoGroup m:type="Edm.Boolean">>false</d:AutoGroup>
        <d:DisableAtNotify m:type="Edm.Boolean">>false</d:DisableAtNotify>
        <d:TermsOfUse m:type="Edm.String" m:null="true"/>
        <d:WebURL m:type="Edm.String" >https://{jam#} [page
246].sapjam.com/groups/sbFeb19ZmYWAkqXtHRoDiq</d:WebURL>
        <d:ContentsVisible m:type="Edm.Boolean">>true</d:ContentsVisible>
        <d:QuestionsVisible m:type="Edm.Boolean">>true</d:QuestionsVisible>
        <d:IdeasVisible m:type="Edm.Boolean">>true</d:IdeasVisible>
        <d:DiscussionsVisible m:type="Edm.Boolean">>true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">>true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">>true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">>true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">>true</d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">>true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">>true</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">>true</d:IsAdmin>
    </m:properties>
</content>
</entry>

```

### Note

The the many navigation URIs for the created group are removed from the response example above to keep the code example concise.

## About POST calls

The unique aspects of using POST calls are:

- You do not set the 'Id' of a resource as a URL parameter.
- You do not use query parameters, except for uploading content, which is shown in the later lesson, [Uploading and downloading files \[page 359\]](#).
- You must include a payload that specifies the GroupTemplate that you want to use to create the Group.
- The success code for a call that creates a resource is 201 Created.
- For a call that creates a resource, a payload listing all properties of the created resource is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form a [POST request for creating a group from a GroupTemplate \[page 295\]](#)?
- Do you know what the [possible responses \[page 296\]](#) are to this request so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a POST call \[page 298\]](#) so that you can easily perform other POST operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Search for a group \[page 299\]](#).

### 8.1.2.3.6 Search for a group

This page of "The Groups and related API" lesson shows you how to search for a Group by using the `[GET] /Groups_Autocomplete` service operation endpoint. This is the third explanation of how to use an OData service operation.

#### Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to create a group \[page 299\]](#).
- Know what the [possible responses \[page 300\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a GET call \[page 303\]](#) so that you can easily perform other GET service operations.

#### Develop the service operation GET request to search for a group

If you want to quickly retrieve the information on a Group whose name or description you know, but whose Id isn't quickly available, try the `[GET] /Groups_Autocomplete` API call to do a string-based search on the Group name.

This section shows the information that must be passed in a `[GET] /Group_Copy` request to create a group by copying an existing group.

1. For all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.


4. Set the URL for your request to be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups_Autocomplete?
Query='{Query}' "
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "?Query='{Query}'" is the service operation parameter of whatever string the currently logged-in user wants to search group titles and group descriptions for.

### **i** Note

Any spaces or special characters in the "Title" or the "Description" must be encoded as described in [RFC2396: Uniform Resource Identifiers \(URI\): Generic Syntax](#) ; for example, spaces must be encoded as "%20".

To summarize these requirements of the request to search for a group in which you set all properties, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed
from here.] */"
-H "Content-Type: application/json" -H "Accept: application/json"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups_Autocomplete?
Query='Customer' "
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!-- [An OAuth2.0 access token has been
removed from here.] -->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups_Autocomplete?
Query='Customer' "
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- For service operations—just as for normal POST operations—if the call successfully creates a resource, you also receive a payload of the full information of the created resource.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "uri": "Groups('vQORd3Mie2QEfyGgmOlJoA')",
          "type": "SAPJam.Group"
        },
        "Id": "vQORd3Mie2QEfyGgmOlJoA",
        "Name": "Customer Group",
        "Description": "A Customer Interaction Group",
        "IsActive": true,
      }
    ]
  }
}
```



```

        "AutoSubscribe": false,
        "Announcement": "",
        "OverviewAsLanding": false,
        "Participation": "full",
        "InvitePolicy": "all",
        "UploadPolicy": "all",
        "ModerationPolicy": false,
        "GroupType": "public_internal",
        "CreatedAt": "/Date(1367324321000)/",
        "LastModifiedAt": "/Date(1367324338000)/",
        "LastActivityAt": "/Date(1378810902000)/",
        "MembersCount": 1,
        "AutoGroup": false,
        "DisableAtNotify": false,
        "TermsOfUse": null,
        "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
vQORd3Mie2QEfyGgm0lJoA",
        "ContentsVisible": true,
        "QuestionsVisible": true,
        "IdeasVisible": true,
        "DiscussionsVisible": true,
        "TasksVisible": true,
        "EventsVisible": true,
        "LinksVisible": true,
        "SubgroupsVisible": true,
        "RecommendationsVisible": true,
        "HasOverview": false,
        "TaskPolicy": "readonly",
        "IsAdmin": false,
        /* [Navigation Properties have been removed from here.] */
    },
    /* [Multiple matching Group entries have been removed from here.] */
  ]}]

```

## Note

In the above example, the many navigation URIs have been cut out, as have multiple matching entries, to keep the code examples concise.

Up to 20 results can be returned, if there are that many matches. If there are more, then the filtering and paging options for returns can be set, which are covered in the next lesson, [OData Query Parameters \[page 323\]](#).

If you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData"
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>Groups_Autocomplete</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups_Autocomplete?
Query=%27Customer%27</id>
  <link rel="self" title="Groups_Autocomplete" href="https://{jam#} [page
246].sapjam.com/api/v1/OData/Groups_Autocomplete?Query=%27Customer%27"/>
  <updated>2016-07-12T15:58:01+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('vQORd3Mie2QEfyGgm0lJoA')</id>
    <title type="text">Groups('vQORd3Mie2QEfyGgm0lJoA')</title>
    <updated>2016-07-12T15:58:01+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Groups('vQORd3Mie2QEfyGgm0lJoA')"
href="Groups('vQORd3Mie2QEfyGgm0lJoA')"/>
  </entry>
</feed>

```

```

    <link
      rel="http://schemas.microsoft.com/ado/2007/08/dataservices/
related/PrimaryGadgetObject"
      type="application/atom+xml;type=entry" title="PrimaryGadgetObject"
      href="Groups('vQORd3Mie2QEfyGgmOlJoA')/PrimaryGadgetObject"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Group" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">vQORd3Mie2QEfyGgmOlJoA</d:Id>
        <d:Name m:type="Edm.String">Customer Group</d:Name>
        <d:Description m:type="Edm.String">Customer Group</
d:Description>
        <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
        <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
        <d:Announcement m:type="Edm.String"/>
        <d:OverviewAsLanding m:type="Edm.Boolean">false</
d:OverviewAsLanding>
        <d:Participation m:type="Edm.String">full</d:Participation>
        <d:InvitePolicy m:type="Edm.String">all</d:InvitePolicy>
        <d:UploadPolicy m:type="Edm.String">all</d:UploadPolicy>
        <d:ModerationPolicy m:type="Edm.Boolean">false</
d:ModerationPolicy>
        <d:GroupType m:type="Edm.String">public_internal</d:GroupType>
        <d:CreatedAt
m:type="Edm.DateTimeOffset">2013-04-30T12:18:41Z</d:CreatedAt>
        <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2013-04-30T12:18:58Z</d:LastModifiedAt>
        <d:LastActivityAt
m:type="Edm.DateTimeOffset">2013-09-10T11:01:42Z</d:LastActivityAt>
        <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
        <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
        <d:DisableAtNotify m:type="Edm.Boolean">false</
d:DisableAtNotify>
        <d:TermsOfUse m:type="Edm.String" m:null="true"/>
        <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/vQORd3Mie2QEfyGgmOlJoA</d:WebURL>
        <d:ContentsVisible m:type="Edm.Boolean">true</
d:ContentsVisible>
        <d:QuestionsVisible m:type="Edm.Boolean">true</
d:QuestionsVisible>
        <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>
        <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">true</
d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">false</d:IsAdmin>
      </m:properties>
    </content>
  </entry>
  <!--[Multiple matching Group entries have been removed from here.]-->
</feed>

```

## i Note

In the above example, the many navigation URIs have been cut out, as have multiple matching entries, to keep the code examples concise.

Up to 20 results can be returned, if there are that many matches. If there are more, then the filtering and paging options for returns can be set, which are covered in the next lesson, [OData Query Parameters \[page 323\]](#).

If your service operation POST request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About service operation GET calls

The unique aspects of using a service operation call that creates a resource are:

- You must set any query parameters that are required for that particular service operation.
- The success code for this particular call, because it is a GET operation, is 200 OK.
- As this call is a GET operation, a payload listing all properties of the matching resources is returned, to a maximum of 20 results. If there are more results, then the filtering and paging options for returns can be set, which are covered in the next lesson, [OData Query Parameters \[page 323\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to create a group \[page 299\]](#)?
- Do you know what the [possible responses \[page 300\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a GET call \[page 303\]](#) so that you can easily perform other GET service operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Update a group \[page 304\]](#).

## 8.1.2.3.7 Update a group

This page of "The Groups and related API" lesson shows you how to update a Group by using the `[PATCH] /Groups('{id}')` endpoint. This is the second explanation of how to use a PATCH operation.

### Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to create a group \[page 304\]](#).
- Know what the [possible responses \[page 307\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a PATCH call \[page 307\]](#) so that you can easily perform other PATCH operations.

### Develop the PATCH request to update a group

As with the creation of a Group, there is only a subset of all properties that can be set in the update operation, `[PATCH] /Groups('{id}')`. Also, for all PATCH operations, note that no payload is returned.

This section shows the information that must be passed in a `[PATCH] /Groups('{id}')` request to update a group.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "PATCH" request.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups('{id}')
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
  - "{Id}", which is the unique identifier of the Group resource that you want to update.
5. Create a payload file that contains the properties and values that you want to update. The Group properties that can be updated are:
    - **Name:** String — The Group's name. Maximum 255 characters.

- **Description:** *String* — The Group's description. Maximum 255 characters.
- **IsActive:** *Boolean* — Whether the Group is set to be visible and usable to members.
- **AutoSubscribe:** *Boolean* — Whether the Group has been configured to have members automatically receive email notifications when new items are posted.
- **Announcement:** *String* — The message that members will see when they first visit the Group.
- **OverviewAsLanding:** *Boolean* — Whether the Overview page has been configured to serve as the Group's landing page.
- **Participation:** *String* — The policy that determines who can participate in the Group. The valid values are "info" (read-only), "expert" (limited), or "full".
- **InvitePolicy:** *String* — The policy that determines who can send invitations to join the Group. The valid values are "all", "followers", or "admins".
- **UploadPolicy:** *String* — The policy that determines who can upload Content to the Group. The valid values are "all", "followers", "internal\_followers", or "admins".
- **ModerationPolicy:** *Boolean* — The policy that determines whether Group administrators must approve member contributions to the Group, including blog posts, documents, photos, videos, feed posts, forum posts, and wiki pages. True if Group administrators must approve contributions; false if not.
- **GroupType:** *String* — The type of the Group. The valid values are "public\_internal", "private\_internal", or "private\_external".
- **DisableAtNotify:** *Boolean* — If set to true, members are not able to use @@notify (notify all) in conversations in the Group.
- **ContentsVisible:** *Boolean* — True if the Group's Content section is set to be visible and available for use to its Members.
- **QuestionsVisible:** *Boolean* — True if the Group's Questions (in Forums) is set to be visible and available for use to its Members.
- **IdeasVisible:** *Boolean* — True if the Group's Ideas (in Forums) is set to be visible and available for use in to its Members.
- **DiscussionsVisible:** *Boolean* — True if the Group's Discussions (in Forums) is set to be visible and available for use to its Members.
- **TasksVisible:** *Boolean* — True if the Group's Tasks feature is set to be visible and available for use to its Members.
- **EventsVisible:** *Boolean* — True if the Group's Events feature is set to be visible and available for use to its Members.
- **LinksVisible:** *Boolean* — True if the Group's Links section is set to be visible and available for use to its Members.
- **SubgroupsVisible:** *Boolean* — True if the Group's Subgroups section is set to be visible and available for use to its Members.
- **RecommendationsVisible:** *Boolean* — True if the Recommendations widget is set to be visible and available in the Group for use by its Members.

Of the above properties, set only those you want to change.

To summarize these requirements of the request to create a group in which you set all properties, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */" \
  -H "Content-Type: application/json" -H "Accept: application/json" \
  -X PATCH "https://{jam#}.sapjam.com/api/v1/OData/Groups('Q0XAzcxqzVVbQ6tQjUqCKu')"
```

```
-d@C:/payloads/payload.json
```

The contents of the `payload.json` file is shown below:

```
{
  "Name": "Customer Assistance Group",
  "Description": "A group to provide customer assistance",
  "IsActive": true,
  "AutoSubscribe": false,
  "Announcement": "Keep our customers happy!",
  "OverviewAsLanding": false,
  "Participation": "full",
  "InvitePolicy": "followers",
  "UploadPolicy": "followers",
  "ModerationPolicy": true,
  "GroupType": "private_external",
  "DisableAtNotify": true,
  "ContentsVisible": true,
  "QuestionsVisible": true,
  "IdeasVisible": true,
  "DiscussionsVisible": true,
  "TasksVisible": true,
  "EventsVisible": true,
  "LinksVisible": true,
  "SubgroupsVisible": true,
  "RecommendationsVisible": true
}
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!-- [An OAuth2.0 access token has been
removed from here.] -->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('QOXAzcqxzVVbQ6tQjUqCKu') "
-d@C:/payloads/payload.xml
```

The contents of the `payload.xml` file is shown below:

```
<entry>
  <content>
    <properties>
      <Name>Customer Support Issue Analysis</Name>
      <Description>A group to track customer requests</Description>
      <IsActive>false</IsActive>
      <AutoSubscribe>true</AutoSubscribe>
      <Announcement>Please be sure to record all customer requests in this
group.</Announcement>
      <OverviewAsLanding>true</OverviewAsLanding>
      <Participation>full</Participation>
      <InvitePolicy>followers</InvitePolicy>
      <UploadPolicy>followers</UploadPolicy>
      <ModerationPolicy>true</ModerationPolicy>
      <GroupType>private_external</GroupType>
      <DisableAtNotify>true</DisableAtNotify>
      <ContentsVisible>true</ContentsVisible>
      <QuestionsVisible>true</QuestionsVisible>
      <IdeasVisible>true</IdeasVisible>
      <DiscussionsVisible>true</DiscussionsVisible>
      <TasksVisible>true</TasksVisible>
      <EventsVisible>true</EventsVisible>
      <LinksVisible>true</LinksVisible>
      <SubgroupsVisible>true</SubgroupsVisible>
      <RecommendationsVisible>true</RecommendationsVisible>
    </properties>
  </content>
</entry>
```

```
</content>
</entry>
```

## Develop the ability to handle either the success or the error response

If your PATCH request is successful, the following information is returned:

- You receive an HTTP success code. For a `PATCH` request, such as this request, the response code is:

```
204 No Content
```

- No payload of information is returned to a successful PATCH operation.

If your PATCH request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About PATCH calls

The unique aspects of using a PATCH call are:

- To update a specific resource, you must set the 'Id' of that resource as a URL parameter.
- You do not use query parameters, except for uploading updated content, as is shown in the later lesson, [Uploading and downloading files \[page 359\]](#).
- You must include a payload that specifies the properties of the resource that you want to update.
- The success code for a PATCH request is always `204 No Content`.
- No payload is returned to a PATCH request.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to create a group \[page 304\]](#)?
- Do you know what the [possible responses \[page 307\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a PATCH call \[page 307\]](#) so that you can easily perform other PATCH operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Delete a group \[page 308\]](#).

### 8.1.2.3.8 Delete a group

This page of "The Groups and related API" lesson shows you how to delete a group by using the `[DELETE] /Groups('{id}')` endpoint. This is the first explanation of how to use a DELETE operation.

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to delete a specific Group \[page 308\]](#).
- Know what the [possible responses \[page 309\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a DELETE call \[page 309\]](#) so that you can easily perform other DELETE operations.

## Develop the DELETE request to delete a specific Group.

This endpoint, `[DELETE] /Groups('{id}')` allows you to delete an existing, specified Group.

The information that must be passed in a `[DELETE] /Groups('{id}')` is as follows:

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. Neither the "Content-Type", nor the "Accept" HTTP headers need to be set.
3. Set your request to be an HTTP "DELETE" method.
4. Set the URL for your request, which will be:

```
https://{jam#} \[page 246\].sapjam.com/api/v1/OData/Groups('{id}')
```



Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Group resource that you want to delete.

To summarize these requirements of the request to delete a Group, here is the curl command:

```
curl -i -H "Authorization: Bearer <!-- [An OAuth2.0 access token has been removed from here.] -->"  
-X DELETE "https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Groups('QOxAzcqxzVVbQ6tQjUqCKu')"
```

## Develop the ability to handle either the success or the error response

If your DELETE request is successful, the following information is returned:

- You receive an HTTP success code. For a DELETE request, such as this request, the response code is:

```
204 No Content
```

- No payload is returned to any DELETE operation.

If your DELETE request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a ContentType that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About DELETE calls

The unique aspects of using a DELETE call are:

- To DELETE a specific resource, you must set the 'Id' of that resource as a URL parameter.
- You do not use query parameters.

- You do not include a request payload.
- The success code for a DELETE request is always 204 No Content.
- No payload is returned to a DELETE request.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to {action to be performed}](#) [\[page 308\]](#)?
- Do you know what the [possible responses](#) [\[page 309\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a DELETE call](#) [\[page 309\]](#) so that you can easily perform other DELETE operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a feed of existing groups](#) [\[page 310\]](#).

### 8.1.2.3.9 Retrieve a feed of existing groups

This page of "The Groups and related API" lesson shows you how to retrieve a feed of Groups by using the collection-valued endpoint, `[GET] /Groups`. The number of groups can be very large on some sites, so this is an excellent segue into "Lesson 3: Manipulating GET feed responses". This is the third explanation of how to use a GET operation, and it is the second explanation of how to use a collection-valued GET operation.

## Learning Objectives

The learning objectives for this page are:

- Know what information is required to form [the request to {action to be performed}](#) [\[page 310\]](#).
- Know what the [possible responses](#) [\[page 311\]](#) are so that you can develop the ability to handle them.
- Understand the [unique aspects of a collection-valued GET call](#) [\[page 316\]](#) so that you can easily perform other collection-valued GET operations.

## Develop the GET request to retrieve a feed of the existing groups

This request, `[GET] /Groups` calls for the information on all of the groups that the currently logged-in user can access. This section shows the information that must be passed in to this request to retrieve the feed of groups.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups
```

Where the variable URL parameter for this call is:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".

### Note

Note that no "Id" parameter is passed in the request. This indicates that a collection-values [GET] request is being made.

To summarize these requirements of the request to {create a group in which you set all properties}, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed from here.] */"
-H "Content-Type: application/json" -H "Accept: application/json"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": {
  "results": [
    {
      "__metadata": {
        "uri": "Groups('X7vRIUosvuBX02hXbLvNag')",
        "type": "SAPJam.Group"
      },
      "Id": "X7vRIUosvuBX02hXbLvNag",
      "Name": "New opportunities",
    }
  ]
}
```

```

    "Description": "A group to list and discuss potential sales
leads",
    "IsActive": true,
    "AutoSubscribe": false,
    "Announcement": null,
    "OverviewAsLanding": false,
    "Participation": "full",
    "InvitePolicy": "followers",
    "UploadPolicy": "followers",
    "ModerationPolicy": false,
    "GroupType": "private_internal",
    "CreatedAt": "/Date(1367962664000)/",
    "LastModifiedAt": "/Date(1367962664000)/",
    "LastActivityAt": "/Date(1389211748000)/",
    "MembersCount": 1,
    "AutoGroup": false,
    "DisableAtNotify": false,
    "TermsOfUse": null,
    "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
X7vRIUosvuBX02hXbLvNag",
    "ContentsVisible": true,
    "QuestionsVisible": true,
    "IdeasVisible": true,
    "DiscussionsVisible": true,
    "TasksVisible": true,
    "EventsVisible": true,
    "LinksVisible": true,
    "SubgroupsVisible": true,
    "RecommendationsVisible": true,
    "HasOverview": false,
    "TaskPolicy": "readonly",
    "IsAdmin": true,
    /* [Navigation Properties have been removed from here.] */
  },
  {
    "_metadata": {
      "uri": "Groups('NVY6beLritm8b2y6nhTgdo')",
      "type": "SAPJam.Group"
    },
    "Id": "NVY6beLritm8b2y6nhTgdo",
    "Name": "Customer Request Decision Group",
    "Description": "A group to track customer requests",
    "IsActive": false,
    "AutoSubscribe": true,
    "Announcement": "Please be sure to record all customer requests
in this group.",
    "OverviewAsLanding": true,
    "Participation": "full",
    "InvitePolicy": "followers",
    "UploadPolicy": "followers",
    "ModerationPolicy": true,
    "GroupType": "private_external",
    "CreatedAt": "/Date(1466702238000)/",
    "LastModifiedAt": "/Date(1466702238000)/",
    "LastActivityAt": "/Date(1466702238000)/",
    "MembersCount": 1,
    "AutoGroup": false,
    "DisableAtNotify": true,
    "TermsOfUse": null,
    "WebURL": "https://{jam#} [page 246].sapjam.com/groups/
NVY6beLritm8b2y6nhTgdo",
    "ContentsVisible": true,
    "QuestionsVisible": true,
    "IdeasVisible": true,
    "DiscussionsVisible": true,
    "TasksVisible": true,
    "EventsVisible": true,
    "LinksVisible": true,

```

```

        "SubgroupsVisible": true,
        "RecommendationsVisible": true,
        "HasOverview": false,
        "TaskPolicy": "readonly",
        "IsAdmin": true,
        /* [Navigation Properties have been removed from here.] */
    },
    /* [18 more Groups entries were cut from here.] */
},
"_next": "Groups?
%24skiptoken=72cq6uhcs2vacpvp2qmgilzo95ww2izhev7ti877ibynad5je9hhxiiclj1rvbic2
ggnurhw3y6tddiiqgrad5jggy2"
}}

```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/odata"
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>Groups</title>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/odata/Groups</id>
    <link rel="self" title="Groups" href="https://
integration3.sapjam.com/api/v1/odata/Groups"/>
    <link rel="next"
        title="Groups?
%24skiptoken=72cq6uhcs2vacpvp2qmgilzo95ww2izhev7ti877ibynad5je9hhxiiclj1rvbic2
ggnurhw3y6tddiiqgrad5jggy2"
        href="Groups?
%24skiptoken=72cq6uhcs2vacpvp2qmgilzo95ww2izhev7ti877ibynad5je9hhxiiclj1rvbic2
ggnurhw3y6tddiiqgrad5jggy2"/>
    <updated>2016-07-15T02:02:11+00:00</updated>
    <entry>
        <id>https://{jam#} [page 246].sapjam.com/api/v1/odata/
Groups('X7vRIUosvubX02hXbLvNag')</id>
        <title type="text">Groups('X7vRIUosvubX02hXbLvNag')</title>
        <updated>2016-07-15T02:02:11+00:00</updated>
        <author>
            <name/>
        </author>
        <link rel="edit" title="Groups('X7vRIUosvubX02hXbLvNag') "
href="Groups('X7vRIUosvubX02hXbLvNag')"/>
        <!--[Navigation Properties have been removed from here.]-->
        <category term="SAPJam.Group" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
        <content type="application/atom+xml">
            <m:properties>
                <d:Id m:type="Edm.String">X7vRIUosvubX02hXbLvNag</d:Id>
                <d:Name m:type="Edm.String">New opportunity</d:Name>
                <d:Description m:type="Edm.String">A group to list and
discuss potential sales leads</d:Description>
                <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
                <d:AutoSubscribe m:type="Edm.Boolean">>false</d:AutoSubscribe>
                <d:Announcement m:type="Edm.String" m:null="true"/>
                <d:OverviewAsLanding m:type="Edm.Boolean">>false</
d:OverviewAsLanding>
                <d:Participation m:type="Edm.String">full</d:Participation>
                <d:InvitePolicy m:type="Edm.String">followers</d:InvitePolicy>
                <d:UploadPolicy m:type="Edm.String">followers</d:UploadPolicy>
                <d:ModerationPolicy m:type="Edm.Boolean">>false</
d:ModerationPolicy>
            </m:properties>
        </content>
    </entry>
</feed>

```

```

        <d:GroupType m:type="Edm.String">private_internal</
d:GroupType>
        <d:CreatedAt
m:type="Edm.DateTimeOffset">2013-05-07T21:37:44Z</d:CreatedAt>
        <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2013-05-07T21:37:44Z</d:LastModifiedAt>
        <d:LastActivityAt
m:type="Edm.DateTimeOffset">2014-01-08T20:09:08Z</d:LastActivityAt>
        <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
        <d:AutoGroup m:type="Edm.Boolean">>false</d:AutoGroup>
        <d:DisableAtNotify m:type="Edm.Boolean">>false</
d:DisableAtNotify>
        <d:TermsOfUse m:type="Edm.String" m:null="true"/>
        <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/X7vRIUosvubX02hXbLvNag</d:WebURL>
        <d:ContentsVisible m:type="Edm.Boolean">>true</
d:ContentsVisible>
        <d:QuestionsVisible m:type="Edm.Boolean">>true</
d:QuestionsVisible>
        <d:IdeasVisible m:type="Edm.Boolean">>true</d:IdeasVisible>
        <d:DiscussionsVisible m:type="Edm.Boolean">>true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">>true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">>true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">>true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">>true</
d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">>true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">>false</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">>true</d:IsAdmin>
    </m:properties>
</content>
</entry>
<entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('NVY6beLritm8b2y6nhTgdo')</id>
    <title type="text">Groups('NVY6beLritm8b2y6nhTgdo')</title>
    <updated>2016-07-15T02:02:11+00:00</updated>
    <author>
        <name/>
    </author>
    <link rel="edit" title="Groups('NVY6beLritm8b2y6nhTgdo') "
href="Groups('NVY6beLritm8b2y6nhTgdo')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Group" scheme="http://
schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
        <m:properties>
            <d:Id m:type="Edm.String">NVY6beLritm8b2y6nhTgdo</d:Id>
            <d:Name m:type="Edm.String">Customer Request Decision Group</
d:Name>
            <d:Description m:type="Edm.String">A group to track customer
requests</d:Description>
            <d:IsActive m:type="Edm.Boolean">>false</d:IsActive>
            <d:AutoSubscribe m:type="Edm.Boolean">>true</d:AutoSubscribe>
            <d:Announcement m:type="Edm.String">Please be sure to record
all customer requests in this group.</d:Announcement>
            <d:OverviewAsLanding m:type="Edm.Boolean">>true</
d:OverviewAsLanding>
            <d:Participation m:type="Edm.String">full</d:Participation>
            <d:InvitePolicy m:type="Edm.String">followers</d:InvitePolicy>
            <d:UploadPolicy m:type="Edm.String">followers</d:UploadPolicy>
            <d:ModerationPolicy m:type="Edm.Boolean">>true</
d:ModerationPolicy>
            <d:GroupType m:type="Edm.String">private_external</
d:GroupType>

```

```

        <d:CreatedAt
m:type="Edm.DateTimeOffset">2016-06-23T17:17:18Z</d:CreatedAt>
        <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2016-06-23T17:17:18Z</d:LastModifiedAt>
        <d:LastActivityAt
m:type="Edm.DateTimeOffset">2016-06-23T17:17:18Z</d:LastActivityAt>
        <d:MembersCount m:type="Edm.Int32">1</d:MembersCount>
        <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
        <d:DisableAtNotify m:type="Edm.Boolean">true</
d:DisableAtNotify>
        <d:TermsOfUse m:type="Edm.String" m:null="true"/>
        <d:WebURL m:type="Edm.String">https://{jam#} [page
246].sapjam.com/groups/NVY6beLritm8b2y6nhTgdo</d:WebURL>
        <d:ContentsVisible m:type="Edm.Boolean">true</
d:ContentsVisible>
        <d:QuestionsVisible m:type="Edm.Boolean">true</
d:QuestionsVisible>
        <d:IdeasVisible m:type="Edm.Boolean">true</d:IdeasVisible>
        <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
        <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
        <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
        <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
        <d:SubgroupsVisible m:type="Edm.Boolean">true</
d:SubgroupsVisible>
        <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
        <d:HasOverview m:type="Edm.Boolean">false</d:HasOverview>
        <d:TaskPolicy m:type="Edm.String">readonly</d:TaskPolicy>
        <d:IsAdmin m:type="Edm.Boolean">true</d:IsAdmin>
    </m:properties>
</content>
</entry>
<!--[18 more Groups entries were cut from here.]-->
</feed>

```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

If your GET request is not successful, one of the following error codes is returned:

- **400 Bad Request:** The server could not understand the request due to malformed syntax.
- **403 Forbidden:** Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the member having insufficient privileges to perform the action.
- **404 Not Found:** The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as `userId`, `groupId`, or `contentId`).
- **405 Method Not Allowed:** The method specified in the Request Line is not allowed for the resource identified by the Request URI.
- **406 Not Acceptable:** The resource identified by the request is only capable of generating response entities which have content characteristics not acceptable according to the accept headers sent in the request. This is the expected response if the HTTP headers request a `ContentType` that cannot be handled by the API or the indicated endpoint of the API.
- **409 Conflict:** There is an internal access conflict to the specified resource.
- **429 Too Many Requests:** The member has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded. See [Rate Limits \[page 254\]](#) to view the established limits.
- **500 Internal Server Error:** The server encountered an unexpected condition, which prevented it from fulfilling the request.
- **501 Not Implemented:** The server does not support the functionality required to fulfill the request.

## About collection-valued GET calls

The unique aspects of using [collection-valued \[page 243\]](#) GET calls are:

- If you want to retrieve the information about all resource of the specified entity type that are available to the currently logged-in user, you set the resource type as usual, but you must not set the 'Id' of a specific resource as a URL parameter.
- You may use query parameters to manage the resources that are returned, although that is shown in the later lesson, [OData Query Parameters \[page 323\]](#).
- As for all GET calls, you do not include a payload in the request.
- The success code for a call that returns a feed of resources is the as for all successful GET calls, 200 OK.
- For a successful collection-valued GET call, a payload listing multiple entries of all properties of the requested resource type is returned.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [the request to retrieve a feed of existing groups \[page 310\]](#) that are available to the currently logged-in user?
- Do you know what the [possible responses \[page 311\]](#) are so that you can develop the ability to handle them?
- Do you understand the [unique aspects of a collection-valued GET call \[page 316\]](#) so that you can easily perform other collection-valued GET operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Groups API Lesson Summary \[page 316\]](#).

## 8.1.2.3.10 Groups API Lesson Summary

This page of "The Groups and related API" lesson provides a summary of this lesson. This lesson really is one of the most important ones, not only because it introduces Groups, the fundamental organizing object in SAP Jam, but also because it introduces the concepts of entities, properties, and navigations, the fundamental API operations of POST, GET, PATCH, and DELETE functions, and OData service operations.

## Learning Summary

This page provides the following summary information of the "The Groups and related API" lesson:

- A table showing a [Summary of API call request and response elements \[page 317\]](#).
- Four tables showing the [Groups and related API properties summaries \[page 318\]](#) (one table for each entity).



- Also, the list of [Available Group and related entities' endpoints \[page 321\]](#) is reproduced from the first page of this lesson.

## Summary of API call request and response elements

The following table presents a summary of the elements that are required in a request for each of the seven different types of call covered in this lesson of "The Groups and related API" lesson.

API call request elements

Request Element	POST calls	GET (entry)	GET (feed)	PATCH call	DELETE call	Service Op (POST)	Service Op (GET)
(Header) Authorization:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
(Header) Content-Type:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
(Header) Accept:	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
HTTP method	POST	GET	GET	PATCH	DELETE	POST	GET
(URL) {jam#}	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>
(URL) {Id}	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
(URL) query options	<input type="checkbox"/>	<input type="checkbox"/>	Optional	<input type="checkbox"/>	<input type="checkbox"/>	varies	varies
payload file	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

The following table presents a summary of the elements that are returned in a response for each of the seven different types of call covered in this lesson of "The Groups and related API" lesson.

API call response elements

Response Element	POST calls	GET (entry)	GET (feed)	PATCH call	DELETE call	Service Op (POST)	Service Op (GET)
Success codes	201 Created	200 OK	200 OK	204 No Content	204 No Content	201 Created or 204 No Content	200 OK
Response payload	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	varies	varies
Error codes	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

## Groups and related API properties summary

There are four OData entities devoted to Groups and to the GroupTemplates that they can be built from. The data that is available in each of these entities is shown in the tables below:

### Note

The first three columns of the following two tables should require no explanation, but some additional information on the last five columns does.

- **Creatable:** [POST] This column has 3 possible entries:
  - **Mandatory:** indicates that this property must be set in a POST creation.
  - **Optional:** indicates that you only need to set this property in a POST creation if you want some other value than the default.
  - **Generated:** indicates that the values of these properties are set by the SAP Jam system and cannot be set through API use.
- **N:** [POST] "Nullable" indicates that the property can be set with an empty value, like this: "".
- **F:** [GET] "Filterable" indicates that you can filter the results of a collection-valued GET request by the values for this property.
- **S:** [GET] "Sortable" indicates that you can sort the results of a collection-valued GET request by the values for this property.
- **U:** [PATCH] "Updatable" indicates that this property can be set in a PATCH request.

### Group properties

Property	Type	Description	Creatable	N	F	S	U
<b>Id</b>	String	The Group's unique ID number, which must be unique within your organization's instance of SAP Jam.	Generated				
<b>Name</b>	String	The Group's name, which must be unique within your organization's instance of SAP Jam. Maximum 255 characters.	Mandatory				
<b>Description</b>	String	The Group's description. Maximum 255 characters. <b>Default:</b> (no content).	Optional				
<b>IsActive</b>	Boolean	Whether the Group is set to be visible and usable to members. Once the Group is activated by changing this property to true, this field is not updatable any more. <b>Default:</b> "true".	Optional				
<b>AutoSubscribe</b>	Boolean	Whether the Group has been configured to have members automatically receive email notifications when new items are posted. <b>Default:</b> "false".	Optional				
<b>Announcement</b>	String	The message that members will see when they first visit the Group. <b>Default:</b> (no content).	Optional				
<b>OverviewAsLanding</b>	Boolean	Whether the Overview page has been configured to serve as the Group's landing page. <b>Default:</b> "false".	Optional				
<b>Participation</b>	String	The policy that determines who can participate in the Group. The valid values are "info" (read-only), "expert" (limited), or "full". <b>Default:</b> "full".	Optional				

Property	Type	Description	Creatable	N	F	S	U
<b>InvitePolicy</b>	String	The policy that determines who can send invitations to join the Group. The valid values are "all", "followers", or "admins". <b>Default:</b> "all".	Optional	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>UploadPolicy</b>	String	The policy that determines who can upload Content to the Group. The valid values are "all", "followers", "internal_followers", or "admins". <b>Default:</b> "all".	Optional	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>ModerationPolicy</b>	Boolean	The policy that determines whether Group administrators must approve member contributions to the Group, including blog posts, documents, photos, videos, feed posts, forum posts, and wiki pages. True if Group administrators must approve contributions; false if not. <b>Default:</b> "false".	Optional	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>GroupType</b>	String	The type of the Group. The valid values are "public_internal", "private_internal", or "private_external". <b>Default:</b> "public_internal".	Optional	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>CreatedAt</b>	DateTimeOffset	The date and time that the Group was created. This will appear in Coordinated Universal Time (UTC) in XML responses, in the form "2014-05-22T23:59:59Z", or in Unix epoch time in JSON responses.	<b>Generated</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>LastModifiedAt</b>	DateTimeOffset	The date and time that the Group configuration was last modified. This will appear in Coordinated Universal Time (UTC) in XML responses, in the form "2014-05-22T23:59:59Z", or in Unix epoch time in JSON responses.	<b>Generated</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>LastActivityAt</b>	DateTimeOffset	The date and time that member submissions last occurred within the Group. This will appear in Coordinated Universal Time (UTC) in XML responses, in the form "2014-05-22T23:59:59Z", or in Unix epoch time in JSON responses.	<b>Generated</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>MembersCount</b>	Int32	The number of members in the Group. The default upon creation is "1".	<b>Generated</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>AutoGroup</b>	Boolean	Whether the group is configured to be automatically populated with members according to preset criteria (based on Member Lists). See <a href="#">"Create and manage member lists"</a> and <a href="#">"Create an auto group"</a> in the <i>SAP Jam Collaboration Administrator Guide</i> for details. <b>Default:</b> "false".	Optional	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>DisableAtNotify</b>	Boolean	If set to true, members are not able to use @@notify (notify all) in conversations in the Group. <b>Default:</b> "false".	Optional	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>
<b>TermsOfUse</b>	String	The full text of the "Terms of Use" that members must agree to in order to access the Group. For members who have accepted the Terms of Use, or for Groups where no Terms of Use has been set, this property will have a null value. <b>Default:</b> (no content).	Optional	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<b>WebURL</b>	String	The URL for the Group landing page.	<b>Generated</b>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Property	Type	Description	Creatable	N	F	S	U
<b>ContentsVisible</b>	Boolean	True if the Group's Content section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>QuestionsVisible</b>	Boolean	True if the Group's Questions (in Forums) is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>IdeasVisible</b>	Boolean	True if the Group's Ideas (in Forums) is set to be visible and available for use in to its Members. <b>Default:</b> "false".	Optional				
<b>DiscussionsVisible</b>	Boolean	True if the Group's Discussions (in Forums) is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>TasksVisible</b>	Boolean	True if the Group's Tasks feature is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>EventsVisible</b>	Boolean	True if the Group's Events feature is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>LinksVisible</b>	Boolean	True if the Group's Links section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>SubgroupsVisible</b>	Boolean	True if the Group's Subgroups section is set to be visible and available for use to its Members. <b>Default:</b> "false".	Optional				
<b>Recommendations-Visible</b>	Boolean	True if the Recommendations widget is set to be visible and available in the Group for use by its Members. <b>Default:</b> "false".	Optional				
<b>HasOverview</b>	Boolean	True if the Group is configured to have an overview page.	<b>Generated</b>				
<b>TaskPolicy</b>	String	The policy that determines who can assign Tasks to others in the Group. The valid values are "readonly", "limited", or "full". <b>Default:</b> "full".	<b>Generated</b>				
<b>IsAdmin</b>	Boolean	Indicates whether the currently logged-in member is an administrator of the group.	<b>Generated</b>				

#### GroupTemplate properties

Property	Type	Description	F	S
<b>Id</b>	String	The GroupTemplate's unique ID number.		
<b>GroupTemplateType</b>	String	The type of the GroupTemplate. The valid values are "system" or "custom".		
<b>Title</b>	String	The GroupTemplate's Title. Maximum 255 characters.		
<b>Description</b>	String	The GroupTemplate's description. <b>Default:</b> (no content).		
<b>Enabled</b>	Boolean	Whether the GroupTemplate is set to be visible and usable to members.		

#### SystemGroupTemplate properties

Property	Type	Description	F	S
<b>Id</b>	String	The SystemGroupTemplate's unique ID number.		

Property	Type	Description	F	S
<b>Title</b>	String	The SystemGroupTemplate's Title.		
<b>Description</b>	String	The SystemGroupTemplate's description.		
<b>Enabled</b>	Boolean	Whether the SystemGroupTemplate is set to be visible and usable to members.		

CustomGroupTemplate properties

Property	Type	Description	F	S	U
<b>Id</b>	String	The CustomGroupTemplate's unique ID number, which must be unique within your organization's instance of SAP Jam.			
<b>Title</b>	String	The CustomGroupTemplate's Title, which must be unique within your organization's instance of SAP Jam. Maximum 255 characters.			
<b>Description</b>	String	The CustomGroupTemplate's description. Maximum 255 characters.			
<b>Enabled</b>	Boolean	Whether the CustomGroupTemplate is set to be visible and usable to members.			
<b>Locale</b>	String	The two-letter country code for the language used in the CustomGroupTemplate.			

## Available Group and related entities' endpoints















The following list of API calls is shown only to give you, the new SAP Jam OData API user, a good sense of what operations are available for the Groups and the various GroupTemplates entities.

While there are many more Groups endpoints that those listed below, these are the ones that are relevant to this basic introduction to the Groups and related entities:

### Note

In the list below, the endpoint signatures link to the [SAP Jam Collaboration API Reference](#). The "(See ...)" notes at the end of the endpoint descriptions link to pages that cover the endpoint in this tutorial.

- [\[POST\] /Groups](#) — Creates a group. (See [Create a group with minimal settings \[page 260\]](#), [Create a group with all options set \[page 266\]](#), and [Create a Group from a template \[page 294\]](#).)
- [\[GET\] /Groups](#) — Retrieves a collection of information on all Groups that the current Member can access. (See [Retrieve a feed of available GroupTemplates \[page 289\]](#).)
- [\[GET\] /Groups\('{id}'\)](#) — Retrieves the information on the specified Group. (See [Retrieve a group's information \[page 277\]](#).)
- [\[PATCH\] /Groups\('{id}'\)](#) — Updates the information on the specified Group. (See [Update an existing Group \[page 304\]](#).)
- [\[DELETE\] /Groups\('{id}'\)](#) — Deletes the specified Group. (See [Delete an existing Group \[page 308\]](#).)
- [\[GET\] /Groups\('{id}'\)/ParentGroup](#) — Retrieves the information on the Parent Group of the specified Group.
- [\[POST\] /Groups\('{id}'\)/SubGroups](#) — Creates a SubGroup of the specified Group.

- `[GET] /Groups('{id}')/SubGroups`  — Retrieves a collection of information on the SubGroups of the specified Group.
- `[GET] /Groups('{id}')/Template`  — Retrieves the information on the GroupTemplate that was used to create the specified Group.
- `[GET] /Groups_Autocomplete`  — Performs a string-based search on Groups that is restricted to the Groups the current logged in Member can access and returns a collection of the Groups that match the specified Query.
  - Parameter: **Query**: The string that the search will be based on.
 (See [Search for a Group \[page 299\]](#).)
- `[POST] /Group_Copy`  — Copies a Group.
  - Parameter: **Id**: The Id of the Group to be copied.
  - Parameter: **NewName**: The new, unique name for the Group that is to be created.
 (See [Copy an existing group \[page 273\]](#).)
- `[GET] /Group_FeedLatestCount`  — Retrieves a count of the FeedEntries posted to a Group's wall ("Feed Updates" section) since the specified last known FeedEntry Id.
  - Parameter: **Id**: The Id of the Group for which you want to retrieve a count of the latest FeedEntries.
  - Parameter: **LatestTopLevelId**: The Id of the last known top-level FeedEntry.
- `[GET] /Groups_IAdminister`  — Retrieves a collection of Groups information of the Groups for which the currently logged-in user is an administrator.
- `[POST] /Group_SaveAsTemplate`  — Saves the structure and content of the indicated Group as a CustomGroupTemplate for future use in setting up a new, similar Group. This will include Forums and blogs, but will exclude FeedEntries, ContentItems (except for blogs), and any Folders structure.
  - Parameter: **Id**: The Id of the Group that you want to save as a CustomGroupTemplate.
  - Parameter: **Title**: The title or name that you want to give to the created CustomGroupTemplate.
  - Parameter: **Description**: The description that you want to give to the created CustomGroupTemplate.
  - Parameter: **Enabled**: Whether you want the created CustomGroupTemplate to be visible and available to Members to create new Groups from.
 (See [Save a group as a CustomGroupTemplate \[page 282\]](#).)
- `[GET] /GroupTemplates`  — Retrieves a collection of all GroupTemplates that are visible to the current user. Note that "GroupTemplates" are a combination of SystemGroupTemplates and CustomGroupTemplates. (See [Retrieve a feed of available GroupTemplates \[page 289\]](#).)
- `[GET] /GroupTemplates(Id='{Id}', GroupTemplateType='{GroupTemplateType}')`  — Retrieves the information on the specified GroupTemplate.
- `[GET] /SystemGroupTemplates`  — Retrieves a collection of all SystemGroupTemplates that are visible to the currently logged-in user.
- `[GET] /SystemGroupTemplates('{id}')`  — Retrieves the information on the specified SystemGroupTemplate.
- `[GET] /CustomGroupTemplates`  — Retrieves a collection of all CustomGroupTemplates that are visible to the currently logged-in user.
- `[GET] /CustomGroupTemplates('{id}')`  — Retrieves the information on the specified CustomGroupTemplate.
- `[PATCH] /CustomGroupTemplates('{id}')`  — Updates the information on the specified CustomGroupTemplate. (See [Update a CustomGroupTemplate \[page 286\]](#).)

## i Note

Only the endpoints above with a "(See ...)" note at the end of the endpoint description are discussed in this tutorial. There should be sufficient examples shown that the use of other endpoints can be easily understood. Also, there are many Groups endpoints that are discussed in other lessons in this tutorial.

## Learning Summary: last chance to review

Do you want to go back to review this material?

- A table showing a [Summary of API call request and response elements \[page 317\]](#).
- Four tables showing the [Groups and related API properties summaries \[page 318\]](#) (one table for each entity).
- Also, the list of [Available Group and related entities' endpoints \[page 321\]](#) is reproduced from the first page of this lesson.

If you are satisfied that you have learned what you need from this entire lesson, continue to the next lesson, [OData Query Parameters \[page 323\]](#).

## 8.1.2.4 OData Query Parameters

Collection-valued GET requests (GET requests that do not specify a particular resource) return a feed (a collection) of entries (resources). The SAP Jam Collaboration OData API will return a maximum of 20 entries in a feed, but there are mechanisms for retrieving results beyond the first 20 entries, for sorting them according to various property values, or for filtering them to show only the entries that you are interested in. You can also limit or expand the amount of information that is returned. This page explains how to manipulate the returned results using a variety of system query options.

## Learning Objectives

Manipulating collection-valued GET response information from the GET request is accomplished in OData using "system query options" that are appended to the API call URL following a question mark (?), and which are each preceded with a dollar sign (\$).

The learning objectives for this page are:

- Know how to [use \\$count to discover how many entries there are \[page 324\]](#).
- Know how to [use \\$top to set the number of entries returned \[page 325\]](#).
- Know how to [use \\$skip to offset the set of entries returned \[page 326\]](#).
- Know how to [use \\$skiptoken to offset the set of entries returned \[page 326\]](#).
- Know how to [use \\$filter to limit the entries returned according to their content \[page 327\]](#).
- Know how to [use \\$orderby to set the criteria that the returned entries are ordered by \[page 329\]](#).
- Know how to [use \\$select to limit and select the data that is shown for the returned entries \[page 331\]](#).

- Know how to [use \\$expand](#) to include details on any property for which there are available navigations [\[page 332\]](#).

## Use `$count` to discover how many entries there are

To find the count of the total number of entries that exist for a particular resource type or entity, use the `$count` system query option, as shown in the following curl command.

i Note

While most of the operations described in this page are "system query options" that are appended to the API call URL following a question mark (?), `$count` is considered to be a "resource path segment", so it is appended to the API call URL with a slash (/).

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been
removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups/$count"
```

This will return only a number (in addition to the curl and HTTP responses, such as the HTTP status code, which will be "200 OK" for a successful response), as is shown below:

```
HTTP/1.1 200 OK
Date: Wed, 22 Apr 2015 15:38:14 GMT
Server: Apache
DataServiceVersion: 2.0
X-RateLimit-Limit: 800
X-RateLimit-Remaining: 795
X-RateLimit-Reset: 1305
ETag: "9a1158154dfa42caddbd0694a4e9bdc8"
Cache-Control: must-revalidate, private, max-age=0
X-Request-Id: fbcdad125220dba54a29a3ba845390cb
X-Runtime: 0.068850
X-Rack-Cache: miss
Status: 200 OK
Vary: Accept-Encoding
Connection: close
Transfer-Encoding: chunked
Content-Type: text/plain; charset=utf-8
Set-Cookie: cookie-bizx2hana=rd13o000000000000000000000ffff0a744f56o80;secure;
path=/
X-XSS-Protection: 1; mode=block
52
```

The result of this particular query, for this particular member in this particular company, is 52 groups. As you can see in the above response, the HTTP status code is shown on line 1, and the "count" is returned on line 21 of this curl response.

i Note

\$count is not available for FeedEntries, Notifications, or GroupMembership API calls.



## Use `$top` to set the number of entries returned

By default, the SAP Jam OData API returns a maximum of 20 resource entries in response to a [collection-valued \[page 243\]](#) request. However, you can use `$top` to set the number of returned entries to any value between 1 and 100. For example, you may want to show a maximum of six results per page in a UI widget, so you might use a call similar to the `curl` command shown below:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?$top=6"
```

This call would return only the first six results, which you could then page through for subsequent sets of six results.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Groups</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24top=6</id>
  <link rel="self" title="Groups"
        keyref="https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24top=6"/>
  <updated>2015-04-22T16:37:04+00:00</updated>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <entry>
    <!-- Entry details removed from here -->
  </entry>
  <!-- Entry details removed from here -->
</feed>
```

For the sake of showing the results in a single screen, the returned information on each entry has been removed (the full response, pretty-printed, was 677 lines long).

To handle the paging for this particular scenario, see the `$skip` section below.

### **i** Note

`$top` is not available for `FeedEntries` or `Notifications` API calls.

## Use `$skip` to offset the set of entries returned

Use the `$skip` system query option to perform paging of results. This is done by setting the number of results to "skip" in each subsequent command. To illustrate this, `$skip` is combined with the `$top` scenario described in the previous section. This usage is shown in the following `curl` command:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?$top=6&$skip=6"
```

This request will show the next six results, from group 7 to group 12, inclusive. Requests for subsequent pages of six results would increase the value of skip accordingly (12, 18, 24, and so on).

No code example of the returned results is shown, as it would appear very much like the example in the `$top` section above.

### i Note

`$skip` is not available for FeedEntries or Notifications API calls.

## Use `$skiptoken` to offset the set of entries returned beginning from the specified entry

As has been stated, the SAP Jam Collaboration API returns a maximum of the 20 resource entries to any collection-valued [page 243] API GET request, but when an SAP Jam OData GET response does return that maximum of 20 entries from a larger collection of resources, the response shows a URL with a `$skiptoken` for calling the next 20 results, which can be seen in lines 9-11 in the following response example:

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Groups</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups</id>
  <link rel="self" title="Groups" keyref="https://{jam#} [page
246].sapjam.com/api/v1/OData/Groups"/>
  <link rel="next"
    title="Groups?
%24skiptoken=6jh0pc7h5052xa7e5g9mzjtcoqvz0n46glvpp073we8gx9nokqozzzgc8i4dmdp9hhe
opawa"
    href="Groups?
%24skiptoken=6jh0pc7h5052xa7e5g9mzjtcoqvz0n46glvpp073we8gx9nokqozzzgc8i4dmdp9hhe
opawa"/>
  <updated>2015-04-22T20:06:10+00:00</updated>
  <entry>
    <!--[Entry details have been removed from here.]-->
  </entry>
  <!--[Multiple Group entries would appear here.]-->
</feed>
```

To call that URL, you would have to append it to the "service root URI", as is shown in the following example curl command:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
      -H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/
      Groups?
$skiptoken=6jh0pc7h5052xa7e5g9mzjtcoqvz0n46g1vpp073we8gx9nokqozzzgc8i4dmdp9hheop
awa"
```

The above request would return the next 20 results of the initially requested resource type.

### Note

\$skiptoken is server-side paging, which cannot be used with the other options in this page that are client-driven requests.

## Use \$filter to limit the entries returned according to their content

The \$filter system query option allows you to limit the returned entries to only those whose properties match the values that you want to see, as is illustrated in the following curl command.

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
      -H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?$filter=Name
      eq 'Customer Support Group'"
```

The above filter would yield something similar to the following response.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Groups</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24filter=Name
+eq+%27Customer+Support+Group%27</id>
  <link rel="self" title="Groups"
        href="https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?
%24filter=Name+eq+%27Customer+Support+Group%27"/>
  <updated>2015-04-24T23:19:57+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups ('BVL0W25SoRV8ZA6XiUxIAA')</id>
    <title type="text">Groups ('BVL0W25SoRV8ZA6XiUxIAA')</title>
    <updated>2015-04-24T23:19:57+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Groups ('BVL0W25SoRV8ZA6XiUxIAA') "
          href="Groups ('BVL0W25SoRV8ZA6XiUxIAA')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Group"
              scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/atom+xml">
      <m:properties>
```

```

    <d:Id m:type="Edm.String">BVLOW25SoRV8ZA6XiUxIAA</d:Id>
    <d:Name m:type="Edm.String">Customer Support Group</d:Name>
    <d:Description m:type="Edm.String">Keeping our customers happy</
d:Description>
    <d:IsActive m:type="Edm.Boolean">true</d:IsActive>
    <d:AutoSubscribe m:type="Edm.Boolean">false</d:AutoSubscribe>
    <d:Announcement m:type="Edm.String"/>
    <d:OverviewAsLanding m:type="Edm.Boolean">false</
d:OverviewAsLanding>
    <d:Participation m:type="Edm.String">full</d:Participation>
    <d:InvitePolicy m:type="Edm.String">followers</d:InvitePolicy>
    <d:UploadPolicy m:type="Edm.String">followers</d:UploadPolicy>
    <d:ModerationPolicy m:type="Edm.Boolean">false</
d:ModerationPolicy>
    <d:GroupType m:type="Edm.String">private_external</d:GroupType>
    <d:CreatedAt m:type="Edm.DateTimeOffset">2013-10-03T17:04:51Z</
d:CreatedAt>
    <d:LastModifiedAt m:type="Edm.DateTimeOffset"
    >2015-02-12T23:57:07Z</d:LastModifiedAt>
    <d:LastActivityAt m:type="Edm.DateTimeOffset"
    >2015-04-24T19:34:57Z</d:LastActivityAt>
    <d:MembersCount m:type="Edm.Int32">3</d:MembersCount>
    <d:AutoGroup m:type="Edm.Boolean">false</d:AutoGroup>
    <d:DisableAtNotify m:type="Edm.Boolean">false</d:DisableAtNotify>
    <d:TermsOfUse m:type="Edm.String" m:null="true"/>
    <d:WebURL m:type="Edm.String"
    >https://{jam#} [page 246].sapjam.com/groups/
BVLOW25SoRV8ZA6XiUxIAA</d:WebURL>
    <d:ContentsVisible m:type="Edm.Boolean">true</d:ContentsVisible>
    <d:QuestionsVisible m:type="Edm.Boolean">true</
d:QuestionsVisible>
    <d:IdeasVisible m:type="Edm.Boolean">false</d:IdeasVisible>
    <d:DiscussionsVisible m:type="Edm.Boolean">true</
d:DiscussionsVisible>
    <d:TasksVisible m:type="Edm.Boolean">true</d:TasksVisible>
    <d:EventsVisible m:type="Edm.Boolean">true</d:EventsVisible>
    <d:LinksVisible m:type="Edm.Boolean">true</d:LinksVisible>
    <d:SubgroupsVisible m:type="Edm.Boolean">true</
d:SubgroupsVisible>
    <d:RecommendationsVisible m:type="Edm.Boolean">true</
d:RecommendationsVisible>
    <d:HasOverview m:type="Edm.Boolean">true</d:HasOverview>
    </m:properties>
  </content>
</entry>
<!--[Multiple Group entries, if there were more matches, would appear
here.]-->
</feed>


```

## i Note

Using `$filter` is currently only supported for the following Entities' properties (where applicable, valid values are shown in parentheses):

- **ContentItem:** ContentItemType ('Page', 'BlogEntry', 'Document', 'Poll', or 'Tool')
- **ContentListItem:** Name
- **ExternalObject:** Exid, ObjectType
- **Folder:** Name
- **FeedEntry:** Read
- **ForumItem:** ForumItemType ("Discussion", "Idea", or "Question")
- **Group:** Id, Name, IsActive, GroupType ("public\_internal", "private\_internal", or "private\_external")
- **GroupExternalObject:** LinkType ("primary", "featured", or "unrelated")
- **GroupMembership:** MemberType ("pending", "site\_admin", "admin", or "member")

- **Idea:** Status ("submitted", "in\_progress", "accepted", "declined", "under\_consideration", or "completed")
- **Notification:** Category ("info", "invitations", "requests", "social", or "tasks"), EventType ("AbuseReported", "AcceptAddedAsAssistant", "AcceptAddedAsMgr", "AcceptAddedAsReport", "AcceptGroupTermsOfUse", "AddedAsAssistant", "AddedAsMgr", "AddedAsReport", "AutoGroupAdmin", "AutoGroupMember", "CommentInDiscussion", "CrossCompanyInviteToGroup", "DiskLimitApproaching", "DiskLimitReached", "EventChange", "EventDeleted", "ExpertiseEndorsementOthers", "ExpertiseEndorsementBestAnswerAdd", "ExpertiseEndorsementBestAnswerView", "FeatureInGroup", "GroupContentPendingApproval", "GroupAccessRequest", "GroupAccessRequestAccepted", "CompanyDistributionListImport", "GroupInvitationListImport", "HiddenByTermsOfUse", "InviteToEvent", "InviteToFollow", "InviteToInactiveGroup", "InviteToGroup", "Like", "MarkComment", "MentionInFeed", "ReceiveKudo", "RequestToBeGroupAdmin", "RejectAddedAsAssistant", "RejectAddedAsMgr", "RejectAddedAsReport", "RejectPendingContent", "RemovedAsAssistant", "RemovedAsMgr", "RemovedAsReport", "ReplyInFeed", "send\_nudge", "SkillNudge", "SubscribedToFeed", "SuggestTopic", "TaskAssigned", "TaskNudge", "user\_new\_features", "AcceptAddedAsFreqColleague", "AcceptAddedAsOccColleague", "account\_updated", "AddedAsFreqColleague", "AddedAsOccColleague", "deal\_updated", "GoalMigrated", "HasAssignedAGoal", "new\_ppl\_location", "people\_visiting\_your\_location", "people\_updated", "Reject", "RejectAddedAsFreqColleague", "RejectAddedAsOccColleague", "RemovedAsFreqColleague", "RemovedAsOccColleague", or "Welcome")
- **Question:** HasBestAnswer
- **Task:** IsOverdue
- **TaskAssignment:** Status ("open", "in\_progress", or "completed")

The OData `$filter` operation supports over a dozen operators. For the full list of operators, and examples of their use, please see the OData v.2.0 specification's ["Filter System Query Option \(\\$filter\)"](#)  documentation.

## Use `$orderby` to set the criteria that the returned entries are ordered by

The `$orderby` system query option allows you to set the criteria by which your results are sorted, although there are limitations in the SAP Jam API on which entities and properties can be used to set the sort order. An example of the `$orderby` usage is shown in the following curl command:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
https://{jam#} [page 246].sapjam.com/api/v1/OData/
Discussions('0MlgLVpePpCntnEWU1kW2')/Comments?$orderby=LastModifiedAt asc"
```

Note the use of "asc" in the `$orderby` request above; this indicates that the results should be returned in "ascending" order. Alternatively, you could use "desc" to have the results returned in "descending" order. The above request would return results similar to the following example response.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://integration2.sapjam.com/api/v1/OData" 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>Discussions('0MlgLVpePpCntnEWU1kW2')/Comments</title>
```

```

<id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
  Discussions('0MlgLVpePpCntnEWU1kW2')/Comments?%24orderby=
%27LastModifiedAt+asc%27</id>
<link rel="self" title="Discussions('0MlgLVpePpCntnEWU1kW2')/Comments"
  href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
  Discussions('0MlgLVpePpCntnEWU1kW2')/Comments?%24orderby=
%27LastModifiedAt+asc%27"/>
<updated>2015-04-24T23:21:30+00:00</updated>
<entry>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
  Comments('Pxx86DCKmoamOqOIF7ojjh')</id>
  <title type="text">Comments('Pxx86DCKmoamOqOIF7ojjh')</title>
  <updated>2015-04-24T23:21:30+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Comments('Pxx86DCKmoamOqOIF7ojjh') "
    href="Comments('Pxx86DCKmoamOqOIF7ojjh')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Comment"
    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">Pxx86DCKmoamOqOIF7ojjh</d:Id>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2014-10-27T22:04:55Z</
d:CreatedAt>
      <d:LastModifiedAt m:type="Edm.DateTimeOffset"
        >2014-10-27T22:04:55Z</d:LastModifiedAt>
      <d:Text m:type="Edm.String">some reply 1</d:Text>
      <d:TextWithPlaceholders m:type="Edm.String">some reply 1</
d:TextWithPlaceholders>
      <d:Liked m:type="Edm.Boolean">false</d:Liked>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:CanDelete m:type="Edm.Boolean">true</d:CanDelete>
    </m:properties>
  </content>
</entry>
<!--[Multiple Comments entries would appear here, with the most recent at
the end of the results ("asc").]-->
</feed>

```

## i Note

Using \$orderby is currently only supported for the following Entities' properties:

- **Comment:** CreatedAt, LastModifiedAt
- **ContentListItem:** Name, LastModifiedAt
- **Discussion:** LastActivity, CommentsCount
- **Idea:** TotalVotes, LastActivity
- **Question:** AnswersCount, LastActivity
- **Task:** DueAt, Priority

## Use `$select` to limit the data that is shown for the returned entries

Use the `$select` option if you want to limit the information that is returned from a GET request. For example, if you are only interested in retrieving the IDs of Groups, you can run the API call shown in the following curl command:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?$select=Id"
```

The returned result would look something like the following:

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Groups</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24select=%27Id
%27</id>
  <link rel="self" title="Groups"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24select=
%27Id%27"/>
  <link rel="next"
    title="Groups?%24select=%27Id%27&%24skiptoken=6jh0pc-etc-opawa"
    href="Groups?%24select=%27Id%27&%24skiptoken=6jh0pc-etc-opawa"/>
  <updated>2015-04-24T18:02:30+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('1JEZuKEYTE8QWIKd2iGhMK')</id>
    <title type="text">Groups('1JEZuKEYTE8QWIKd2iGhMK')</title>
    <updated>2015-04-24T18:02:30+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Groups('1JEZuKEYTE8QWIKd2iGhMK') "
      href="Groups('1JEZuKEYTE8QWIKd2iGhMK')"/>
    <category term="SAPJam.Group"
      scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">1JEZuKEYTE8QWIKd2iGhMK</d:Id>
      </m:properties>
    </content>
  </entry>
  <!--[Multiple Group entries would appear here.]-->
</feed>
```

As you can see, the usual set of 30 properties that is normally shown for each Group, is reduced to only the property that you "selected".

### i Note

The `$select` option can be run on GET requests that specify a single resource, not only on unspecified GET requests that return a feed of entries.

## Use `$expand` to include details on any available navigation

The `$expand` system query option, as illustrated in the following curl command, allows you to get the information on a feed of groups, with the added (expanded) information on the Member who created each Group.

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?
$expand=Creator"
```

The output from such a request can be extremely large: with all of the navigation properties included (prior to removing them for this example), the example was nearly 200 lines long for just one "group-with-creator" entry. For the default of 20 entries per feed page, this would be a page of nearly 4000 lines.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Groups</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24expand=
%27Creator%27</id>
  <link rel="self" title="Groups"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/Groups?%24expand=
%27Creator%27"/>
  <link rel="next"
    title="Groups?%24expand=%27Creator%27&%24skiptoken=6jh0pc--etc--opawa"
    href="Groups?%24expand=%27Creator%27&%24skiptoken=6jh0pc--etc--opawa"/>
  <updated>2015-04-24T19:58:59+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Groups('1JEZuKEYTE8QWIKd2iGhMK')</id>
    <title type="text">Groups('1JEZuKEYTE8QWIKd2iGhMK')</title>
    <updated>2015-04-24T19:58:59+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Groups('1JEZuKEYTE8QWIKd2iGhMK') "
      href="Groups('1JEZuKEYTE8QWIKd2iGhMK')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Creator"
      type="application/atom+xml;type=entry" title="Creator"
      href="Groups('1JEZuKEYTE8QWIKd2iGhMK')/Creator">
      <m:inline type="application/atom+xml;type=entry">
        <entry>
          <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('3carMxYCZJP2mskM4MR5OP')</id>
          <title type="text">Members('3carMxYCZJP2mskM4MR5OP')</title>
          <updated>2015-04-24T19:58:59+00:00</updated>
          <author>
            <name/>
          </author>
          <link rel="edit" title="Members('3carMxYCZJP2mskM4MR5OP') "
            href="Members('3carMxYCZJP2mskM4MR5OP')"/>
          <!--[Navigation Properties have been removed from here.]-->
          <category term="SAPJam.Member"
            scheme="http://schemas.microsoft.com/ado/2007/08/
dataservices/scheme"/>
          <content type="application/atom+xml">
            <m:properties>
```



```

d:Id>
    <d:Id m:type="Edm.String">3carMxYCZJP2mskM4MR5OP</
    <d:FirstName m:type="Edm.String">John</d:FirstName>
    <d:LastName m:type="Edm.String">Doe</d:LastName>
    <d:Nickname m:type="Edm.String" m:null="true"/>
    <d:Title m:type="Edm.String">Software Developer</
d:Title>
    <d:Email m:type="Edm.String">john.doe@example.com</
d:Email>
    <d:FullName m:type="Edm.String">John Doe</d:FullName>
    <d:Role m:type="Edm.String">company</d:Role>
    <d:IsFollowing m:type="Edm.Boolean">>false</
d:IsFollowing>
    </m:properties>
  </content>
</entry>
</m:inline>
</link>
<!--[Navigation Properties have been removed from here.]-->
<category term="SAPJam.Group"
  scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <!--[Group Properties have been removed from here.]-->
    </m:properties>
  </content>
</entry>
<!--[Multiple Group entries, with expanded Creator entries, would appear
here.]-->
</feed>

```

Because of the increased risk of stress on the responsiveness of the SAP Jam service from `$expand` calls, the following limitations are imposed.

### Note

There can be at most 3 `$expand` operations in one API call. Using `$expand` on a [collection-valued \[page 243\]](#) navigation of a collection is not allowed.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to [use `\$count` to discover how many entries there are \[page 324\]](#)?
- Do you know how to [use `\$top` to set the number of entries returned \[page 325\]](#)?
- Do you know how to [use `\$skip` to offset the set of entries returned \[page 326\]](#)?
- Do you know how to [use `\$skiptoken` to offset the set of entries returned \[page 326\]](#)?
- Do you know how to [use `\$filter` to limit the entries returned according to their content \[page 327\]](#)?
- Do you know how to [use `\$orderby` to set the criteria that the returned entries are ordered by \[page 329\]](#)?
- Do you know how to [use `\$select` to limit and select the data that is shown for the returned entries \[page 331\]](#)?
- Do you know how to [use `\$expand` to include details on any property for which there are available navigations \[page 332\]](#)?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Uploading and downloading files \[page 359\]](#).

## 8.1.2.5 The Members and related API

SAP Jam Collaboration member information is imported from your member identity provider service, so there are no create, update, or delete methods for members, only retrieve calls. There are create, update, and delete calls for some of the navigations.

### Learning Objectives

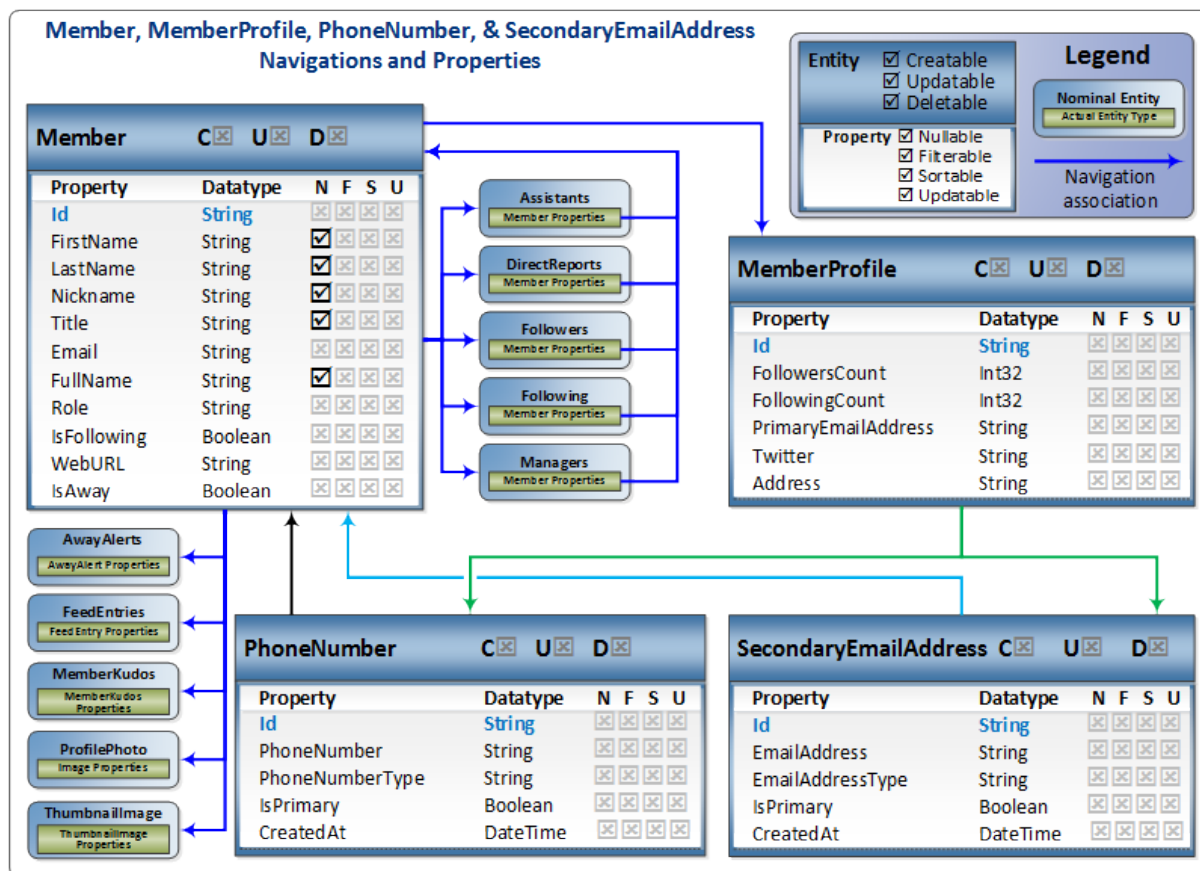
The learning objectives for this page are:

- Get an initial sense of the [properties and navigations \[page 334\]](#) that make up, and are available to, these entities.
- Get an initial sense of the [endpoints \[page 335\]](#) that are available to use and manage these entities.

### Member and related entities' properties and navigations

The following diagram is provided to give you, the new SAP Jam OData API user, a good sense of what data (or "properties") the depicted entities represent. The diagram also shows the navigations from each of the entities covered. Navigations can be thought of as URL extensions from the base entity to closely related entities. For example, in the URL `[POST] /Members('{id}')/FeedEntries`, "FeedEntries" is the navigation. The API call in this example creates a FeedEntry on the specified Member's "wall".

There are four OData entities devoted to member information. The data provided by these entities, and the navigations to other entities is shown in the diagram below and the descriptions that follow it:



Member, MemberProfile, PhoneNumber, and SecondaryEmailAddress entities' properties and navigations

## Available Endpoints

The following list of API calls is shown only to give you, the new SAP Jam OData API user, a good sense of what operations are available for the Member, MemberProfile, PhoneNumbers, and SecondaryEmailAddress entities.

### Note

In the list below, the endpoint signatures link to the SAP Jam Collaboration API Reference. The "(See ...)" notes at the end of the endpoint descriptions link to pages that cover the endpoint in this tutorial.

- [\[GET\] /Members\('{id}'\)](#) — Retrieves the information on the specified Member. (See [Retrieve a specific user's Member information \[page 340\]](#).)
- [\[GET\] /Members\('{id}'\)/Assistants](#) — Retrieves the Member information on the Assistants of the specified Member.
- [\[GET\] /Members\('{id}'\)/AwayAlerts](#) — Retrieves any AwayAlerts for the specified Member.
- [\[GET\] /Members\('{id}'\)/DirectReports](#) — Retrieves the Member information on the DirectReports of the specified Member.

- `[POST] /Members('{id}')/FeedEntries` — Creates a FeedEntries on the "wall" of the specified Member.
- `[GET] /Members('{id}')/FeedEntries` — Retrieves the FeedEntries of the specified Member.
- `[GET] /Members('{id}')/Followers` — Retrieves the Member information on the Followers of the specified Member. (See [Retrieve a feed of Members who are followers of the specified Member \[page 348\]](#) and [Retrieve a count of Members who are followers of the specified Member \[page 351\]](#).)
- `[GET] /Members('{id}')/Following` — Retrieves the Member information on those that the specified Member is Following. (See [Retrieve a feed of the Members the specified Member is following \[page 344\]](#).)
- `[POST] /Members('{id}')/$links/Following` — Sets the specified Member as Following another Member. (See [Set the current user as following another \[page 342\]](#).)
- `[DELETE] /Members('{id}')/$links/Following('{id1}')` — Deletes the link that sets the specified Member as Following another specified Member. (See [Unset the current user as following another \[page 353\]](#).)
- `[GET] /Members('{id}')/Managers` — Retrieves a collection of Member information on the Managers of the specified Member.
- `[GET] /Members('{id}')/MemberProfile` — Retrieves the MemberProfile of the specified Member.
- `[GET] /Self` — Retrieves the Member information of the current Member. (See [Retrieve the current user's Member information \[page 337\]](#).)
- `[GET] /Members_Autocomplete` — Performs a string-based search on the Members of the specified Group that the currently logged-in Member can access and returns a collection of Member information on the Members that match the Query. This is the endpoint to use when implementing at mention functionality for a Group, as it only shows users from the specified Group.
- `[GET] /Members_FindByEmail` — Retrieves a Member's information by specifying their email address.
- `[GET] /Members_FindBySyncId` — Retrieves the Member information by specifying their SuccessFactors Platform Identity Provider (IdP) sync ID.
- `[GET] /Member_FollowsFeedLatestCount` — Retrieves a count of the unread messages in the Member's "Follows" feed.
- `[GET] /Member_ProfileFeedLatestCount` — Retrieves a count of the unread messages in the Member's "Profile" feed.
- `[GET] /MemberProfiles('{id}')` — Retrieves the specified MemberProfile. (See [Get a Member's complete profile information \[page 355\]](#).)
- `[GET] /MemberProfiles('{id}')/PhoneNumbers` — Retrieves a collection of the PhoneNumbers associated with the specified MemberProfile.
- `[GET] /MemberProfiles('{id}')/SecondaryEmailAddresses` — Retrieves a collection of the SecondaryEmailAddresses associated with the specified MemberProfile.
- `[GET] /PhoneNumbers('{id}')` — Retrieves the specified PhoneNumber.
- `[GET] /PhoneNumbers('{id}')/Member` — Retrieves the information on the Member associated with the specified PhoneNumber.
- `[GET] /SecondaryEmailAddresses('{id}')` — Retrieves the specified SecondaryEmailAddress.
- `[GET] /SecondaryEmailAddresses('{id}')/Member` — Retrieves the information on the Member associated with the specified SecondaryEmailAddress.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you have an initial sense of the [properties and navigations \[page 334\]](#) that make up, and are available to, these entities?
- Do you have an initial sense of the [endpoints \[page 335\]](#) that are available to use and manage these entities?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve the current user's Member information \[page 337\]](#).

### 8.1.2.5.1 Retrieve the current user's Member information

This page describes how to use the `[GET] /Self` service operation endpoint to get the currently logged-in user's Member information.

## Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to get the currently logged-in user's Member information \[page 337\]](#).
- Know what the [possible responses \[page 338\]](#) are so that you can develop the ability to handle them.

## Develop the service operation GET request to get the currently logged-in user's Member information

This section shows the information that must be passed in a `[GET] /Self` request to get the currently logged-in user's Member information.

### i Note

These instructions for a GET request that retrieves a single resource that {action to be performed} are for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a group's information \[page 277\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all OData API POST, GET, and PATCH requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.

3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Self
```

Where the variable URL parameter for this call is:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- Optionally, set any [system query options \[page 323\]](#) that you want to use to modify the feed of results that are returned.

To summarize these requirements of the request to {action to be performed}, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
-H "Content-Type: application/json" -H "Accept: application/json"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Self"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'  
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/Self"
```

## Develop the ability to handle either the success or the error response

If your GET service operation request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- For any successful GET operation, you also receive a payload of either the full information about the requested resource or a feed of entries that are available to the currently logged-in user for the requested entity type.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": { "results": {  
  " _metadata": {  
    "uri": "Members('2qFbDP80jDZ21W8QzZehaM')",  
    "type": "SAPJam.Member"  
  },  
  "Id": "2qFbDP80jDZ21W8QzZehaM",  
  "FirstName": "Randy",  
  "LastName": "Winger",  
  "Nickname": null,  
  "Title": "Chief Information Officer",  
  "Email": "rwinger@example.com",  
  "FullName": "Randy Winger",  
  "Role": "company",  
  "IsFollowing": false,  
  "WebURL": "https://{jam#} [page 246].sapjam.com/profile/wall/  
2qFbDP80jDZ21W8QzZehaM",  
  "IsAway": false,  
  /* [Navigation Properties have been removed from here.] */  
}}
```

```
}}}
```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/odata/
Members('h9y1PDglLPZ0VtqHRtHq5A')</id>
  <title type="text">Members('h9y1PDglLPZ0VtqHRtHq5A')</title>
  <updated>2016-09-03T01:15:07+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Members('h9y1PDglLPZ0VtqHRtHq5A') "
    href="Members('h9y1PDglLPZ0VtqHRtHq5A') "/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Member"
    scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
  <content type="application/xml">
    <m:properties>
      <d:Id m:type="Edm.String">h9y1PDglLPZ0VtqHRtHq5A</d:Id>
      <d:FirstName m:type="Edm.String">Carla</d:FirstName>
      <d:LastName m:type="Edm.String">Grant</d:LastName>
      <d:Nickname m:type="Edm.String" m:null="true"/>
      <d:Title m:type="Edm.String">VP, Sales</d:Title>
      <d:Email m:type="Edm.String">cgrant@example.com</d:Email>
      <d:FullName m:type="Edm.String">Carla Grant</d:FullName>
      <d:Role m:type="Edm.String">company</d:Role>
      <d:IsFollowing m:type="Edm.Boolean">>false</d:IsFollowing>
      <d:WebURL m:type="Edm.String"
        >https://{jam#} [page 246].sapjam.com/profile/wall/
h9y1PDglLPZ0VtqHRtHq5A</d:WebURL>
      <d:IsAway m:type="Edm.Boolean">>false</d:IsAway>
    </m:properties>
  </content>
</entry>
```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to get the currently logged-in user's Member information \[page 337\]](#)?
- Do you know what the [possible responses \[page 338\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a specific user's Member information \[page 340\]](#).

## 8.1.2.5.2 Retrieve a specific user's Member information

This page shows you how to retrieve a specified user's Member information by using the `[GET] /Members('{Id}')` endpoint.

### Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to retrieve a specified user's Member information \[page 340\]](#).
- Know what the [possible responses \[page 341\]](#) are so that you can develop the ability to handle them.

### Develop the GET request to retrieve the information on a single Member

This section shows the information that must be passed in a `[GET] /Members('{Id}')` request to retrieve the information on the specified Member.

#### Note

These instructions for a GET request that retrieves a single resource that {action to be performed} are for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a group's information \[page 277\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all GET requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/[GET] /Members('{Id}')
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Member resource whose information you want to retrieve.

To summarize these requirements of the request to retrieve the information on the specified Member, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
      -H "Content-Type: application/json" -H "Accept: application/json"  
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members('h9y1PDg1LPZ0VtqHRtHq5A')"
```



Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('2qFbDP80jDZ21W8QzZehaM') "
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request that creates a resource, such as this request, the response code is:

```
200 OK
```

- For a resource-specific GET operation, you also receive a payload of the a feed of entries available to the currently logged-in user for the requested entity.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "Members('h9y1PDglLPZ0VtqHRtHq5A')",
        "type": "SAPJam.Member"
      },
      "Id": "h9y1PDglLPZ0VtqHRtHq5A",
      "FirstName": "Carla",
      "LastName": "Grant",
      "Nickname": null,
      "Title": "VP, Sales",
      "Email": "cgrant@example.com",
      "FullName": "Carla Grant",
      "Role": "company",
      "IsFollowing": true,
      "WebURL": "https://{jam#} [page 246].sapjam.com/profile/wall/h9y1PDglLPZ0VtqHRtHq5A",
      "IsAway": false,
      /* [Navigation Properties have been removed from here.] */
    }
  }
}
```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('2qFbDP80jDZ21W8QzZehaM')</id>
  <title type="text">Members('2qFbDP80jDZ21W8QzZehaM')</title>
  <updated>2016-09-03T02:31:56+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Members('2qFbDP80jDZ21W8QzZehaM') "
    href="Members('2qFbDP80jDZ21W8QzZehaM') "/>
</entry>
```

```

<!--[Navigation Properties have been removed from here.]-->
<category term="SAPJam.Member"
  scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
  <content type="application/xml">
    <m:properties>
      <d:Id m:type="Edm.String">2qFbDP80jDZ21W8QzZehaM</d:Id>
      <d:FirstName m:type="Edm.String">Randy</d:FirstName>
      <d:LastName m:type="Edm.String">Winger</d:LastName>
      <d:Nickname m:type="Edm.String" m:null="true"/>
      <d:Title m:type="Edm.String">Chief Information Officer</d:Title>
      <d:Email m:type="Edm.String">rwinger@example.com</d:Email>
      <d:FullName m:type="Edm.String">Randy Winger</d:FullName>
      <d:Role m:type="Edm.String">company</d:Role>
      <d:IsFollowing m:type="Edm.Boolean">>false</d:IsFollowing>
      <d:WebURL m:type="Edm.String"
        >https://{jam#} [page 246].sapjam.com/profile/wall/
        2qFbDP80jDZ21W8QzZehaM</d:WebURL>
      <d:IsAway m:type="Edm.Boolean">>false</d:IsAway>
    </m:properties>
  </content>
</entry>

```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to retrieve a specified user's Member information \[page 340\]](#)?
- Do you know what the [possible responses \[page 341\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Set the current user as following another \[page 342\]](#).

### 8.1.2.5.3 Set the current user as following another

The "following" function is set by configuring the current user as following another user by using the [POST] /Members ('{Id}')/\$links/Following endpoint.

## Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to set the current user as following another user \[page 343\]](#).

- Know what the [possible responses \[page 344\]](#) are so that you can develop the ability to handle them.

## Develop a service operation POST request to set the current user as following another user

This section shows the information that must be passed in a [POST] /Members('{Id}')/\$links/Following request to set the current user as following another user.

1. Set the following HTTP headers:
  - Authorization (See [Authentication and Authorization API \[page 395\]](#) for details.)
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
2. Set your request to be an HTTP "POST" method.
3. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('{Id}')/$links/Following
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the currently logged-in user who want to follow another Member.

4. Set the URI of the Member that the user wants to follow in a payload file.

To summarize these requirements of the request to set the current user as following another user, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H "Content-Type: application/json" -H "Accept: application/json"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('{Id}')/$links/Following"
-d@/home/johndoe/Documents/payloads/payload.json
```

This next example for the JSON work flow shows the contents of the payload file:

```
{
  "uri" : "Members('3carMxYCZJP2mskM4MR5OP')"
}
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H "Content-Type: application/atom+xml"
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('{Id}')/$links/Following"
-d@C:/Users/johndoe/Documents/payloads/payload.xml
```

This next example for the XML work flow shows the contents of the payload file:

```
<uri>Members('2qFbDP80jDZ21W8QzZehaM')</uri>
```

## Develop the ability to handle either the success or the error response

If your POST service operation request is successful, the following information is returned:

- You receive an HTTP success code. For a POST request that does not create a resource, such as this request, the response code is:

```
204 No Content
```

- No payload is returned to any POST operation that does not create a resource.

If your request is not successful, an error code is returned. See HTTP error codes.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to set the current user as following another user \[page 343\]](#)?
- Do you know what the [possible responses \[page 344\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a feed of the Members the specified Member is following \[page 344\]](#).

### 8.1.2.5.4 Retrieve a feed of the Members the specified Member is following

Retrieve a feed of Members that the specified Member is following by using the [GET] `/Members('{id}')/Following` endpoint.

## Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to retrieve a feed of Members that the specified Member is following \[page 345\]](#).
- Know what the [possible responses \[page 346\]](#) are so that you can develop the ability to handle them.

## Develop the GET request to retrieve a feed of Members that the specified Member is following

This section shows the information that must be passed in a `[GET] /Members ('{Id}') /Following` request to retrieve a feed of Members that the specified Member is following.

### Note

These instructions for a GET request that retrieves a feed of entries is for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a feed of available GroupTemplates \[page 289\]](#) and [Retrieve a feed of existing groups \[page 310\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all GET requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Members ('{Id}') /Following
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Member from whom you want to retrieve a feed of Member resources for those he or she is following.
- Optionally, set any [system query options \[page 323\]](#) that you want to use to modify the feed of results that are returned.

To summarize these requirements of the request to retrieve a feed of Members that the specified Member is following, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
-H "Content-Type: application/json" -H "Accept: application/json"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members ('h9y1PDglLPZ0VtqHRtHq5A') /Following"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'  
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"  
"https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members ('h9y1PDglLPZ0VtqHRtHq5A') /Following"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- For a collection-valued GET operation, you also receive a payload that shows a feed of Members that the specified Member is following.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": { "results": [
  {
    "__metadata": {
      "uri": "Members('0XXbopPMwCj7BuZovKYxz1')",
      "type": "SAPJam.Member"
    },
    "Id": "0XXbopPMwCj7BuZovKYxz1",
    "FirstName": "Perry ",
    "LastName": "Johnson",
    "Nickname": null,
    "Title": "Account Manager",
    "Email": "perry.johnson@example.com",
    "FullName": "Perry Johnson",
    "Role": "company",
    "IsFollowing": false,
    "WebURL": "https://{jam#} [page 246].sapjam.com/profile/wall/0XXbopPMwCj7BuZovKYxz1",
    "IsAway": false,
    /* [Navigation Properties have been removed from here.] */
  },
  /* [Multiple Member entries have been removed from here.] */
]}
```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://integration3.sapjam.com/api/v1/OData" 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>Members('2qFbDP80jDZ21W8QzZehaM')/Following</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('2qFbDP80jDZ21W8QzZehaM')/Following</id>
  <link rel="self" title="Members('2qFbDP80jDZ21W8QzZehaM')/Following"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('2qFbDP80jDZ21W8QzZehaM')/Following"/>
  <updated>2016-09-04T15:52:28+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('3carMxYCZJP2mskM4MR5OP')</id>
    <title type="text">Members('3carMxYCZJP2mskM4MR5OP')</title>
    <updated>2016-09-04T15:52:28+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Members('3carMxYCZJP2mskM4MR5OP') "
      href="Members('3carMxYCZJP2mskM4MR5OP')"/>
    <!--[Navigation Properties have been removed from here.]-->
```

```

    <category term="SAPJam.Member"
      scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/xml">
      <m:properties>
        <d:Id m:type="Edm.String">3carMxYCZJP2mskM4MR5OP</d:Id>
        <d:FirstName m:type="Edm.String">William</d:FirstName>
        <d:LastName m:type="Edm.String">Hansen</d:LastName>
        <d:Nickname m:type="Edm.String" m:null="true"/>
        <d:Title m:type="Edm.String">VP, Information Technology</
d:Title>
        <d:Email m:type="Edm.String">whansen@example.com</d:Email>
        <d:FullName m:type="Edm.String">William Hansen</d:FullName>
        <d:Role m:type="Edm.String">company</d:Role>
        <d:IsFollowing m:type="Edm.Boolean">true</d:IsFollowing>
        <d:WebURL m:type="Edm.String"
>https://{jam#} [page 246].sapjam.com/profile/wall/
3carMxYCZJP2mskM4MR5OP</d:WebURL>
        <d:IsAway m:type="Edm.Boolean">false</d:IsAway>
      </m:properties>
    </content>
  </entry>
  <!--[Multiple Member entries have been removed from here.]-->
</feed>

```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise. Also, multiple entries have been removed.

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to retrieve a feed of Members that the specified Member is following \[page 345\]](#)?
- Do you know what the [possible responses \[page 346\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a feed of Members who are followers of the specified Member \[page 348\]](#).

## 8.1.2.5.5 Retrieve a feed of Members who are followers of the specified Member

This page describes how to retrieve a feed of followers of the specified Member by using the `[GET] /Members('{id}')/Followers` endpoint.

### Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to retrieve a feed of followers of the specified Member \[page 348\]](#).
- Know what the [possible responses \[page 349\]](#) are so that you can develop the ability to handle them.

### Develop the GET request to retrieve a feed of followers of the specified Member

This section shows the information that must be passed in a `[GET] /Members('{id}')/Followers` request to retrieve a feed of Members who are followers of the specified Member.

#### i Note

These instructions for a GET request that retrieves a feed of entries is for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a feed of available GroupTemplates \[page 289\]](#) and [Retrieve a feed of existing groups \[page 310\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all GET requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('{id}')/Followers
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Member for whom you want to retrieve a feed of Members that are his or her followers.
- Optionally, set any [system query options \[page 323\]](#) that you want to use to modify the feed of results that are returned.



To summarize these requirements of the request to retrieve a feed of Members who are followers of the specified Member, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
      -H "Content-Type: application/json" -H "Accept: application/json"
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('h9y1PDglLPZ0VtqHRtHq5A')/Followers"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
      -H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('h9y1PDglLPZ0VtqHRtHq5A')/Followers"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- For this collection-valued GET operation, you also receive a payload of Members information for those that are followers of the specified Member.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{ "d": { "results": [
  {
    "__metadata": {
      "uri": "Members('0XXbopPMwCj7BuZovKYxz1')",
      "type": "SAPJam.Member"
    },
    "Id": "0XXbopPMwCj7BuZovKYxz1",
    "FirstName": "Perry ",
    "LastName": "Johnson",
    "Nickname": null,
    "Title": "Account Executive",
    "Email": "perry.johnson@example.com",
    "FullName": "Perry Johnson",
    "Role": "company",
    "IsFollowing": false,
    "WebURL": "https://{jam#} [page 246].sapjam.com/profile/wall/
0XXbopPMwCj7BuZovKYxz1",
    "IsAway": false,
    /* [Navigation Properties have been removed from here.] */
  },
  /* [Multiple Member entries have been removed from here.] */
]}
```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```

<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData"
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>Members ('2qFbDP80jDZ21W8QzZehaM')/Followers</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members ('2qFbDP80jDZ21W8QzZehaM')/Followers</id>
  <link rel="self" title="Members ('2qFbDP80jDZ21W8QzZehaM')/Followers"
href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members ('2qFbDP80jDZ21W8QzZehaM')/Followers"/>
  <updated>2016-09-04T18:27:00+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members ('3carMxYCZJP2mskM4MR5OP')</id>
    <title type="text">Members ('3carMxYCZJP2mskM4MR5OP')</title>
    <updated>2016-09-04T18:27:00+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Members ('3carMxYCZJP2mskM4MR5OP') "
href="Members ('3carMxYCZJP2mskM4MR5OP') "/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Member"
scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
    <content type="application/xml">
      <m:properties>
        <d:Id m:type="Edm.String">3carMxYCZJP2mskM4MR5OP</d:Id>
        <d:FirstName m:type="Edm.String">William</d:FirstName>
        <d:LastName m:type="Edm.String">Hansen</d:LastName>
        <d:Nickname m:type="Edm.String" m:null="true"/>
        <d:Title m:type="Edm.String">VP, Information Technology</
d:Title>
        <d:Email m:type="Edm.String">whansen@example.com</d:Email>
        <d:FullName m:type="Edm.String">William Hansen</d:FullName>
        <d:Role m:type="Edm.String">company</d:Role>
        <d:IsFollowing m:type="Edm.Boolean">true</d:IsFollowing>
        <d:WebURL m:type="Edm.String"
>https://{jam#} [page 246].sapjam.com/profile/wall/
3carMxYCZJP2mskM4MR5OP</d:WebURL>
        <d:IsAway m:type="Edm.Boolean">false</d:IsAway>
      </m:properties>
    </content>
  </entry>
  <!--[Multiple Member entries have been removed from here.]-->
</feed>

```

Note that in the above example, the many navigation URIs have been removed to keep the code example concise.

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to retrieve a feed of followers of the specified Member \[page 348\]](#)?
- Do you know what the [possible responses \[page 349\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Retrieve a count of Members who are followers of the specified Member \[page 351\]](#).

## 8.1.2.5.6 Retrieve a count of Members who are followers of the specified Member

This page describes how to retrieve a count of followers of the specified Member by using the `[GET] /Members('{id}')/Followers` endpoint with a `$count` query option.

### Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to retrieve a count of followers of the specified Member \[page 351\]](#).
- Know what the [possible responses \[page 352\]](#) are so that you can develop the ability to handle them.

### Develop the GET request to retrieve a count of followers of the specified Member

This section shows the information that must be passed in a `[GET] /Members('{id}')/Followers` endpoint with a `$count` query option to retrieve a count of followers of the specified Member.

#### i Note

These instructions for a GET request that retrieves a count of followers of the specified Member is for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a feed of available Group Templates \[page 289\]](#) and [Retrieve a feed of existing groups \[page 310\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all GET requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('{id}')/Followers?$count
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".

- "{Id}", which is the unique identifier of the Member for whom you want to retrieve a feed of Followers.
- Optionally, set any [system query options \[page 323\]](#) that you want to use to modify the feed of results that are returned.

To summarize these requirements of the request to retrieve a feed of Member entries that are followers of the specified Member, here is the curl command:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('h9ylPDglLPZ0VtqHRtHq5A')/Followers?$count"
```

### i Note

As a \$count has been requested, neither the format of the request nor the format of the return are relevant.

### i Note

This same use of the \$expand system query option can be used with the [GET] /Members('{id}')/Following endpoint, described in [Retrieve a feed of the Members the specified Member is following \[page 344\]](#) to retrieve a count of Members that are following the specified member.

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request, such as this request, the response code is:

```
200 OK
```

- Normally, for a collection-valued GET operation, you also receive a payload that shows a feed of entries available to the currently logged-in user for the requested entity, but as the \$count query option was set, the returned payload is an integer indicating the number of Members who are followers of the specified Member, so the returned payload will look like this:

```
837
```

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to retrieve a count of followers of the specified Member \[page 351\]](#)?
- Do you know what the [possible responses \[page 352\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Unset the current user as following another \[page 353\]](#).

## 8.1.2.5.7 Unset the current user as following another

This page describes how to delete the link that sets one user as following another by using the `[DELETE] /Members('{id}')/$links/Following('{id1}')` endpoint.

### Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to delete the link that sets one user as following another \[page 353\]](#).
- Know what the [possible responses \[page 354\]](#) are so that you can develop the ability to handle them.

### Develop the DELETE request to delete the link that sets one user as following another

This section shows the information that must be passed in a `[DELETE] /Members('{id}')/$links/Following('{id1}')` request to delete the link that sets one user as following another.

#### Note

These instructions for a DELETE request that removes {action to be performed} are for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Delete a group \[page 308\]](#).

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. Set your request to be an HTTP "DELETE" method.
3. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/{generalizedEndpoint}
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Member who is following another member.
- "{Id1}", which is the unique identifier of the Member that is being followed.

To summarize these requirements of the request to delete the link that sets one user as following another, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"  
-X DELETE "https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members('3carMxYCZJP2mskM4MR5OP')/$links/Following('2qFbDP80jDZ21W8QzZehaM') "
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'  
-X DELETE "https://{jam#} [page 246].sapjam.com/api/v1/OData/  
Members('3carMxYCZJP2mskM4MR5OP')/$links/Following('h9y1PDglLPZ0VtqHRtHq5A') "
```

## Develop the ability to handle either the success or the error response

If your DELETE request is successful, the following information is returned:

- You receive an HTTP success code. For a DELETE request, such as this request, the response code is:

```
204 No Content
```

- No payload is returned to any DELETE operation.

If your request is not successful, an error code is returned. See HTTP error codes.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to delete the link that sets one user as following another \[page 353\]](#)?
- Do you know what the [possible responses \[page 354\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Get a Member's complete profile information \[page 355\]](#).

## 8.1.2.5.8 Get a Member's complete profile information

This page describes how to retrieve all of the details of a Member's profile, including their additional phone numbers and secondary email addresses by using the `[GET] /MemberProfiles('{Id}')` endpoint with expansions of their `PhoneNumbers` and `SecondaryEmailAddresses`.

### Learning Objectives

The learning objectives for this page are:

- Know how to form [the request to retrieve all of the details of a Member's profile \[page 355\]](#).
- Know what the [possible responses \[page 356\]](#) are so that you can develop the ability to handle them.

### Develop the GET request to retrieve all of the details of a Member's profile

This section shows the information that must be passed in a `[GET] /MemberProfiles('{Id}')` request with expansions of their `PhoneNumbers` and `SecondaryEmailAddresses` to retrieve all of the details of a Member's profile, including their additional phone numbers and secondary email addresses.

#### Note

These instructions for a GET request that retrieves a single resource of the complete details of a Member's profile are for a somewhat experienced audience. If you would like more details on how to make such an API request, see [Retrieve a group's information \[page 277\]](#). Also, this page discusses the use of the [OData system query option, \\$expand \[page 332\]](#) to include details from the available navigations of `PhoneNumbers` and `SecondaryEmailAddresses`.

1. Like all OData API requests, you must set the "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. For all GET requests, you must set:
  - The "Content-Type" HTTP header to indicate the format of the information being sent in this request.
  - The "Accept" HTTP header to indicate the format that you will accept in the response.
3. Set your request to be an HTTP "GET" method.
4. Set the URL for your request, which will be:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/MemberProfiles('{Id}')?
$expand=PhoneNumbers,SecondaryEmailAddresses
```

Where the variable URL parameters for this call are:

- "{jam#}", which is your organization's SAP Jam data center, such as "jam4".
- "{Id}", which is the unique identifier of the Member whose profile information you want to retrieve.
- "?\$expand=PhoneNumbers,SecondaryEmailAddresses", which are the navigations to related entities with additional profile information that you want to retrieve as well.

To summarize these requirements of the request to retrieve the information on the specified {entity\_type}, here is the curl command, if you choose to use the JSON format:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H "Content-Type: application/json" -H "Accept: application/json"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/MemberProfiles('3carMxYCZJP2mskM4MR5OP')?$expand=PhoneNumbers,SecondaryEmailAddresses"
```

Alternatively, here is the curl command, if you choose to use the XML format:

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H "Content-Type: application/atom+xml" -H "Accept: application/atom+xml"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/MemberProfiles('3carMxYCZJP2mskM4MR5OP')?$expand=PhoneNumbers,SecondaryEmailAddresses"
```

## Develop the ability to handle either the success or the error response

If your GET request is successful, the following information is returned:

- You receive an HTTP success code. For a GET request that creates a resource, such as this request, the response code is:

```
200 OK
```

- For a resource-specific GET operation, you also receive a payload of the a feed of entries available to the currently logged-in user for the requested entity.

If you set the Accept header to "application/json", the returned payload will look like this:

```
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "MemberProfiles('3carMxYCZJP2mskM4MR5OP')",
        "type": "SAPJam.MemberProfile"
      },
      "Id": "3carMxYCZJP2mskM4MR5OP",
      "FollowersCount": 56,
      "FollowingCount": 55,
      "PrimaryEmailAddress": "whansen@example.com",
      "Twitter": null,
      "Address": null,
      "PhoneNumbers": {
        "results": [
          {
            "__metadata": {
              "uri": "PhoneNumbers('BFYNcElyPQeWDHEQp087RU')",
              "type": "SAPJam.PhoneNumber"
            },
            "Id": "BFYNcElyPQeWDHEQp087RU",
            "PhoneNumber": "212-555-1212",
            "PhoneNumberType": "Home",
            "IsPrimary": false,
            "CreatedAt": "/Date(1358194588000)/",
            "Member": {
              "__deferred": {
                "uri": "PhoneNumbers('BFYNcElyPQeWDHEQp087RU')/Member"
              }
            }
          }
        ]
      },
      "SecondaryEmailAddresses": {
        "results": [
          {
            "__metadata": {
              "uri": "SecondaryEmailAddresses('tkMrQz3kiFkHajeOYOCda')",
              "type": "SAPJam.SecondaryEmailAddress"
            }
          }
        ]
      }
    }
  }
}
```



```

    },
    "Id": "tktMrQz3kiFkHajeOYOCda",
    "EmailAddress": "william.hansen212@someemailservice.com",
    "EmailAddressType": "Home",
    "IsPrimary": false,
    "CreatedAt": "/Date(1376499557000)/",
    "Member": {"__deferred": {"uri":
"SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda')/Member"}}
    }]]
}}}

```

Alternatively, if you set the Accept header to "application/atom+xml", the returned payload will look like this:

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
MemberProfiles('3carMxYCZJP2mskM4MR5OP')</id>
  <title type="text">MemberProfiles('3carMxYCZJP2mskM4MR5OP')</title>
  <updated>2016-09-05T20:07:04+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="MemberProfiles('3carMxYCZJP2mskM4MR5OP') "
    href="MemberProfiles('3carMxYCZJP2mskM4MR5OP')"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
PhoneNumbers"
    type="application/atom+xml;type=feed" title="PhoneNumbers"
    href="MemberProfiles('3carMxYCZJP2mskM4MR5OP')/PhoneNumbers">
  <m:inline type="application/atom+xml;type=feed">
    <feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData"
      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>MemberProfiles('3carMxYCZJP2mskM4MR5OP')</title>
      <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
MemberProfiles('3carMxYCZJP2mskM4MR5OP')?%24expand=%27PhoneNumbers
%2CSecondaryEmailAddresses%27</id>
      <link rel="self"
title="MemberProfiles('3carMxYCZJP2mskM4MR5OP') "
        href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
MemberProfiles('3carMxYCZJP2mskM4MR5OP')?%24expand=%27PhoneNumbers
%2CSecondaryEmailAddresses%27"/>
      <updated>2016-09-05T20:07:04+00:00</updated>
      <entry>
        <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
PhoneNumbers('BFYNcElyPQeWDHEQp087RU')</id>
        <title
type="text">PhoneNumbers('BFYNcElyPQeWDHEQp087RU')</title>
        <updated>2016-09-05T20:07:04+00:00</updated>
        <author>
          <name/>
        </author>
        <link rel="edit"
title="PhoneNumbers('BFYNcElyPQeWDHEQp087RU') "
          href="PhoneNumbers('BFYNcElyPQeWDHEQp087RU')"/>
        <link rel="http://schemas.microsoft.com/ado/2007/08/
dataservices/related/Member"
          type="application/atom+xml;type=entry" title="Member"
          href="PhoneNumbers('BFYNcElyPQeWDHEQp087RU')/Member"/>
        <category term="SAPJam.PhoneNumber"
          scheme="http://schemas.microsoft.com/ado/2007/08/
dataservices/scheme"/>

```

```

        <content type="application/xml">
            <m:properties>
                <d:Id m:type="Edm.String">BFYNcElyPQeWDHEQp087RU</
d:Id>
                <d:PhoneNumber m:type="Edm.String">212-555-1212</
d:PhoneNumber>
                <d:PhoneNumberType m:type="Edm.String">Home</
d:PhoneNumberType>
                <d:IsPrimary m:type="Edm.Boolean">>false</
d:IsPrimary>
                <d:CreatedAt m:type="Edm.DateTimeOffset"
>2013-01-14T20:16:28Z</d:CreatedAt>
            </m:properties>
        </content>
    </entry>
</feed>
</m:inline>
</link>
<link
    rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
SecondaryEmailAddresses"
    type="application/atom+xml;type=feed" title="SecondaryEmailAddresses"
    href="MemberProfiles('3carMxYCZJP2mskM4MR5OP')/
SecondaryEmailAddresses">
    <m:inline type="application/atom+xml;type=feed">
        <feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData"
            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>MemberProfiles('3carMxYCZJP2mskM4MR5OP')</title>
            <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
MemberProfiles('3carMxYCZJP2mskM4MR5OP')?%24expand=%27PhoneNumbers
%2CSecondaryEmailAddresses%27</id>
            <link rel="self"
title="MemberProfiles('3carMxYCZJP2mskM4MR5OP') "
            href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
MemberProfiles('3carMxYCZJP2mskM4MR5OP')?%24expand=%27PhoneNumbers
%2CSecondaryEmailAddresses%27"/>
            <updated>2016-09-05T20:07:04+00:00</updated>
            <entry>
                <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda')</id>
                <title
type="text">SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda')</title>
                <updated>2016-09-05T20:07:04+00:00</updated>
                <author>
                    <name/>
                </author>
                <link rel="edit"
title="SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda') "
                href="SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda')"/>
                <link rel="http://schemas.microsoft.com/ado/2007/08/
dataservices/related/Member"
                    type="application/atom+xml;type=entry" title="Member"
                href="SecondaryEmailAddresses('tktMrQz3kiFkHajeOYOCda')/Member"/>
                <category term="SAPJam.SecondaryEmailAddress"
                    scheme="http://schemas.microsoft.com/ado/2007/08/
dataservices/scheme"/>
                <content type="application/xml">
                    <m:properties>
                        <d:Id m:type="Edm.String">tktMrQz3kiFkHajeOYOCda</
d:Id>
                        <d:EmailAddress m:type="Edm.String"

```

```

                                >william.hansen212@someemailservice.com</
d:EmailAddress>
                                <d:EmailAddressType m:type="Edm.String">Home</
d:EmailAddressType>
                                <d:IsPrimary m:type="Edm.Boolean">>false</
d:IsPrimary>
                                <d:CreatedAt m:type="Edm.DateTimeOffset"
                                >2013-08-14T16:59:17Z</d:CreatedAt>
                                </m:properties>
                                </content>
                                </entry>
                                </feed>
                                </m:inline>
                                </link>
                                <category term="SAPJam.MemberProfile"
                                scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
                                scheme"/>
                                <content type="application/xml">
                                <m:properties>
                                <d:Id m:type="Edm.String">3carMxYCZJP2mskM4MR5OP</d:Id>
                                <d:FollowersCount m:type="Edm.Int32">56</d:FollowersCount>
                                <d:FollowingCount m:type="Edm.Int32">55</d:FollowingCount>
                                <d:PrimaryEmailAddress m:type="Edm.String"
                                >whansen@example.com</d:PrimaryEmailAddress>
                                <d:Twitter m:type="Edm.String" m:null="true"/>
                                <d:Address m:type="Edm.String" m:null="true"/>
                                </m:properties>
                                </content>
                                </entry>

```

If your request is not successful, an error code is returned. See [HTTP error codes \[page 254\]](#).

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [the request to retrieve all of the details of a Member's profile \[page 355\]](#)?
- Do you know what the [possible responses \[page 356\]](#) are so that you can develop the ability to handle them?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Uploading and downloading files \[page 359\]](#).

### 8.1.2.6 Uploading and downloading files

The SAP Jam Collaboration OData API entities that are used to manage files (BLOBs) are `ContentItem`, `Image` (for which the role alias of `ProfilePhoto` can be used), `ThumbnailImage`, `FeedEntryImage` (for which the role alias of `PreviewImage` can be used), and `ThumbnailKudoImage`. For any POST, GET, or PATCH call, you can affect only the information about the file, or you can create, retrieve, or update the actual file. This page describes how the API calls must be set to create, retrieve, or update the files.

## Learning Objectives

The learning objectives for this page are:

- Know which [HTTP headers \[page 360\]](#) must be set to handle file uploads and downloads.
- Know that you must [Set the \\$value query option \[page 361\]](#) to handle file uploads and downloads.
- Understand the [example curl commands \[page 361\]](#) well enough that you can easily perform other file upload and download operations.

For the GET and PATCH calls, you must set all of the following to have the call download or upload the file ([BLOB \[page 243\]](#)):

1. The "Authorization" HTTP header, which must include either the calling user's OAuth1.0a protocol parameters, or their OAuth2.0 access token, is required for all SAP Jam OData API calls. Details on how to develop this mechanism are covered in the [Authentication and Authorization API \[page 395\]](#).
2. The Content-Type HTTP header must be set to the proper type for the file that is to be uploaded or downloaded.
3. The Slug HTTP header must be set to specify the name of the file that will be, or that currently is, used by SAP Jam.
4. The \$value URL parameter must be set to indicate that the "raw value" of the resource (the BLOB, or file) to be uploaded or downloaded, rather than just the information about the file.

Also, in any PATCH request, you must provide a link to the file from which it can be uploaded.

## Set the required HTTP headers

Please refer to the *Authorization API* section for more information.

Header Field	Use	Description
<b>Slug:</b>	Mandatory for file uploads or downloads	"Slug" is an HTTP header whose presence in a POST to a Collection constitutes a request by the client to use the header's value as part of any URIs that would normally be used to retrieve the to-be-created Entry or Media Resources. Effectively, this is the string that you want to appear, or that does appear, as the name of the file in SAP Jam. To upload a file in a PATCH operation that you want to appear in SAP Jam as "profilePhoto.png", or to download a file in a GET operation that currently appears in SAP Jam as "profilePhoto.png", you must set <code>Slug: profilePhoto</code> .

Header Field	Use	Description
<b>Content-Type:</b>	Mandatory for file uploads or downloads	<p>The value of the Content-Type header must be set to the MIME type of the BLOB (file) type that you want to download or upload. Valid options include:</p> <ul style="list-style-type: none"> <li>• <code>text/html;type=wiki</code></li> <li>• <code>text/html;type=blog</code></li> <li>• <code>text/html</code></li> <li>• <code>image/png</code></li> <li>• <code>image/jpg</code></li> <li>• <code>video/mp4</code></li> <li>• <code>application/vnd.ms-powerpoint</code></li> <li>• <code>application/vnd.openxml-formats-officedocument.presentationml.presentation</code></li> </ul> <p>For a complete list of possible MIME types, please see <a href="http://www.iana.org/assignments/media-types/media-types.xhtml">http://www.iana.org/assignments/media-types/media-types.xhtml</a>. Note that SAP Jam has tested a sub-set of this list, covering the most commonly used formats; however, we make no guarantee of support for every MIME type.</p> <p>Example: <code>ContentType: image/png</code></p>

Also, in any PATCH request, you must provide a link to the file from which it can be uploaded.

## Set the `$value` query option

This ensures that the "raw resource" (the file being referred to) is uploaded or downloaded, rather than the information about the file.

URL Segment	Required	Description
<b><code>\$value</code></b>	Mandatory for file uploads or downloads	Indicates that the request should act upon the "raw value" of the resource: the actual content item.

## Example curl commands

### JSON: A Member updates his or her profile image

```
curl -i -H 'Authorization: OAuth <!--[The OAuth1.0a protocol parameters have been removed from here.]-->'
-H 'Content-Type: image/png' -H 'Slug: JaneDoePortrait'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('2qFbDP80jDZ21W8QzZehaM')/ProfilePhoto/$value"
-d "@/home/janedoe/graphics/JaneDoePortrait.png"
```

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## XML: Updating a file

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: image/png' -H 'Slug: JohnDoeIdPhoto'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('3carMxYCZJP2mskM4MR5OP')/ContentItems/$value"
-d "@C:/Users/johndoe/Documents/JohnDoeIdPhoto.jpg"
```

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## JSON: Update a wiki using an attached file

```
curl -i -H 'Authorization: OAuth <!--[The OAuth1.0a protocol parameters have been removed from here.]-->'
-H 'Content-Type: text/html;type=wiki' -H 'Slug: mySqlTutorialWiki'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/ContentItems(Id='ceoIVCtYaLSDuPNgxfOSvt',ContentItemType='Page')/$value"
-d "@C:/Users/johndoe/Documents/WikiSource/mySqlTutorialWiki.html"
```

Note that the HTML file must meet the requirements described in [Supported HTML tags \[page 390\]](#).

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## XML: Update a wiki using an attached file

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: text/html;type=wiki' -H 'Slug: JavaScriptGuidelines'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/ContentItems(Id='bTk9lZEoylmGM3nqEEucTE',ContentItemType='Page')/$value"
-d "@/home/janedoe/wikis/JavaScriptGuidelines.html"
```

Note that the HTML file must meet the requirements described in [Supported HTML tags \[page 390\]](#).

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## JSON: Updating a blog with an HTML file as the payload

```
curl -i -H 'Authorization: OAuth <!--[The OAuth1.0a protocol parameters have been removed from here.]-->'
-H 'Content-Type: text/html;type=blog' -H 'Slug: DatabaseManagementBlog'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/ContentItems(Id='X7vRIUosvuBX02hXbLvNag',ContentItemType='BlogEntry')/$value"
-d "@C:/Users/johndoe/Documents/BlogSource/DatabaseManagementBlog.html"
```

Note that the HTML file must meet the requirements described in [Supported HTML tags \[page 390\]](#).

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## XML: Updating a blog with an HTML file as the payload

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: text/html;type=blog' -H 'Slug: CodingBestPractices'
-X PATCH "https://{jam#} [page 246].sapjam.com/api/v1/OData/ContentItems(Id='Gvaz2JfroQBc9imgnIBvkP',ContentItemType='BlogEntry')/$value"
-d "@/home/janedoe/blogs/CodingBestPractices.html"
```

Note that the HTML file must meet the requirements described in [Supported HTML tags \[page 390\]](#).

As a PATCH operation, no payload is returned, and the success code will be: 204 No Content.

## Endpoints that upload or download files

The SAP Jam Collaboration OData API includes the following endpoints that will upload or download files:

- `[GET] /Groups('{id}')/ProfilePhoto/$value` — Downloads the actual image used as the ProfilePhoto of the specified Group.
- `[PATCH] /Groups('{id}')/ProfilePhoto/$value` — Uploads an update for the image used as the ProfilePhoto of the specified Group.
- `[GET] /Members('{id}')/ProfilePhoto/$value` — Downloads the image used as the ProfilePhoto of the specified Member.
- `[PATCH] /Members('{id}')/ProfilePhoto/$value` — Uploads an updated version of the image used as the ProfilePhoto of the specified Member.
- `[GET] /Members('{id}')/ThumbnailImage/$value` — Downloads the ThumbnailImage for the specified Member.
- `[GET] /ContentItems(Id='{Id}', ContentItemType='{ContentItemType}')/$value` — Downloads the specified ContentItem.
- `[PATCH] /ContentItems(Id='{Id}', ContentItemType='{ContentItemType}')/$value` — Updates (uploads) the specified ContentItem.
- `[GET] /ContentListItems(Id='{Id}', ContentListItemType='{ContentListItemType}')/ContentItem/$value` — Downloads the ContentItem that is associated with the specified ContentListItem.
- `[PATCH] /ContentListItems(Id='{Id}', ContentListItemType='{ContentListItemType}')/ContentItem/$value` — Uploads an update of the ContentItem that is associated with the specified ContentListItem.
- `[GET] /MirrorLinks(Id='{Id}', MirrorLinkType='{MirrorLinkType}')/ContentItem/$value` —
- `[PATCH] /MirrorLinks(Id='{Id}', MirrorLinkType='{MirrorLinkType}')/ContentItem/$value` —
- `[GET] /Images('{id}')/$value` — Downloads the image file for the specified Image.
- `[PATCH] /Images('{id}')/$value` — Uploads an update of the specified Image.
- `[GET] /ThumbnailImages(Id='{Id}', ThumbnailImageType='{ThumbnailImageType}')/$value` — Downloads the image used for the specified ThumbnailImage.
- `[GET] /Comments('{id}')/ThumbnailImage/$value` — Downloads the image used as the ThumbnailImage in the specified Comment.
- `[GET] /FeedEntries('{id}')/PreviewImage/$value` — Downloads the image of the FeedEntryImage that is used as the PreviewImage of the specified FeedEntry.
- `[GET] /FeedEntries('{id}')/ThumbnailImage/$value` — Downloads the ThumbnailImage of the user who posted the specified FeedEntry.
- `[GET] /FeedEntryImages('{id}')/$value` — Retrieves the image of the specified FeedEntryImage.
- `[GET] /Kudos(Id='{Id}', KudoType='{KudoType}')/ThumbnailKudoImage/$value` — Retrieves the ThumbnailKudoImage image file used in the specified Kudos.
- `[GET] /ThumbnailKudoImages(Id='{Id}', ThumbnailImageType='{ThumbnailImageType}')/$value` — Downloads the image used for the specified ThumbnailKudoImage.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know which [HTTP headers \[page 360\]](#) must be set to handle file uploads and downloads?
- Do you know that you must [Set the \\$value query option \[page 361\]](#) to handle file uploads and downloads?
- Do you understand the [example curl commands \[page 361\]](#) well enough that you can easily perform other file upload and download operations?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Using @-mention in the API \[page 364\]](#).

## 8.1.2.7 Using @-mention in the API

This section describes how at-mentions are handled in the SAP OData API.

### Learning Objectives

The learning objectives for this page are:

- Know what information is required to set an [@mention placeholder \[page 364\]](#).
- Understand the example, [\[GET\] /Comments\('{key}'\) \[page 365\]](#).
- Understand the example, [\[GET\] /Comments\('{key}'\)/AtMentions \[page 366\]](#).
- Understand the example, [\[POST\] /Discussions\('{key}'\)/Comments \[page 368\]](#).
- Understand the example, [\[GET\] /WallComments\('{key}'\) \[page 369\]](#).
- Understand the example, [\[GET\] /WallComments\('{key}'\)/AtMentions \[page 370\]](#).
- Understand the example, [\[POST\] /Ideas\('{key}'\)/Posts \[page 372\]](#).
- Understand the problem in the bad request example, [\[POST\] /Discussions\('{key}'\)/Comments \(1\) \[page 374\]](#).
- Understand the problem in the second bad request example, [\[POST\] /Discussions\('{key}'\)/Comments \(2\) \[page 374\]](#).

### @mention placeholders

For the text properties of entities that can contain at-mentions, the property must be set as `<propertyName>WithPlaceholders`; for example:

- The `Comment` entity's `Text` property must be set as `TextWithPlaceholders`.
- The `WallComment` entity's `Comment` property must be set as `CommentWithPlaceholders`.

The `<propertyName>WithPlaceholders` properties are read only.

In these properties, the at-mentions are replaced with placeholders.



The placeholder is numbered (zero-indexed) to indicate which Member appears in which position, for example:

- Could @@m{0} and @@m{1} please review this.

Placeholder numbers do not always appear incrementally ordered from left to right, so both of the following examples are possible:

- Could @@m{0} and @@m{1} please review this.
- Could @@m{1} and @@m{0} please review this.

Placeholders match the regular expression `/@@{\d+}/`.

Placeholders are always unique, so the following example is not possible:

- Could @@m{0} and @@m{0} please review this.

At-mention duplicates are possible in a collection with multiple mentions of the same member in the text.

The links count and placeholder numbers must correspond in post. Order is important as it denotes the order that the at-mentions appear in the feed text.

## [GET] /Comments('{key}')

### Curl command (JSON)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/json'
"https://[{jam#}] [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')"
```

### JSON Response

```
{
  "d": {
    "results": {
      "_metadata": {
        "uri": "Comments('3Ag5RzNRVPiA0To99NCPCM')",
        "type": "SAPJam.Comment"
      },
      "Id": "3Ag5RzNRVPiA0To99NCPCM",
      "CreatedAt": "/Date(1415172839000)/",
      "LastModifiedAt": "/Date(1415172839000)/",
      "Text": "hello @admin and @test102 ",
      "TextWithPlaceholders": "hello @@m{0} and @@m{1} ",
      "Liked": false,
      "LikesCount": 0,
      "CanDelete": true,
      /* [Navigation Properties have been removed from here.] */
    }
  }
}
```

### Curl command (XML)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://[{jam#}] [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')"
```

## XML Response

```
<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')</id>
  <title type="text">Comments('3Ag5RzNRVPiA0To99NCPCM')</title>
  <updated>2014-11-05T07:35:50+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Comments('3Ag5RzNRVPiA0To99NCPCM') "
href="Comments('3Ag5RzNRVPiA0To99NCPCM')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Comment" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">3Ag5RzNRVPiA0To99NCPCM</d:Id>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2014-11-05T07:33:59Z</
d:CreatedAt>
      <d:LastModifiedAt m:type="Edm.DateTimeOffset">2014-11-05T07:33:59Z</
d:LastModifiedAt>
      <d:Text m:type="Edm.String">hello @admin and @test102 </d:Text>
      <d:TextWithPlaceholders m:type="Edm.String">hello @@m{0} and @@m{1}
</d:TextWithPlaceholders>
      <d:Liked m:type="Edm.Boolean">false</d:Liked>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:CanDelete m:type="Edm.Boolean">true</d:CanDelete>
    </m:properties>
  </content>
</entry>
```

## [GET] /Comments('{key}')/AtMentions

### Curl command (JSON)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/json'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions"
```

### JSON Response

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "uri": "Members('R4EkdiTleXTPpBkTiWyi8Q')",
          "type": "SAPJam.Member"
        },
        "Id": "R4EkdiTleXTPpBkTiWyi8Q",
        "FirstName": "admin",
        "LastName": "test",
        "Nickname": null,
        "Title": "dev",
        "Email": "admin@test.com",
        "FullName": "admin test",
        "Role": "company",
      }
    ]
  }
}
```

```

        "IsFollowing": false,
        /* [Navigation Properties have been removed from here.] */
    },
    /* [Multiple AtMentions entries would appear here.] */
}]]

```

## Curl command (XML)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/atom+xml'
'https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions'

```

## XML Response

```

<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions</id>
  <link rel="self" title="Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('3Ag5RzNRVPiA0To99NCPCM')/AtMentions"/>
  <updated>2014-11-05T07:37:21+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('R4EkdiTleXTPpBkTiWyi8Q')</id>
    <title type="text">Members('R4EkdiTleXTPpBkTiWyi8Q')</title>
    <updated>2014-11-05T07:37:21+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Members('R4EkdiTleXTPpBkTiWyi8Q')"
href="Members('R4EkdiTleXTPpBkTiWyi8Q')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Member" scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">R4EkdiTleXTPpBkTiWyi8Q</d:Id>
        <d:FirstName m:type="Edm.String">admin</d:FirstName>
        <d:LastName m:type="Edm.String">test</d:LastName>
        <d:Nickname m:type="Edm.String" m:null="true"/>
        <d>Title m:type="Edm.String">dev</d>Title>
        <d>Email m:type="Edm.String">admin@test.com</d>Email>
        <d:FullName m:type="Edm.String">admin test</d:FullName>
        <d:Role m:type="Edm.String">company</d:Role>
        <d:IsFollowing m:type="Edm.Boolean">false</d:IsFollowing>
      </m:properties>
    </content>
  </entry>
  <!--[Multiple AtMentions entries would appear here.]-->
</feed>

```

## [POST] /Discussions('{key}')/Comments

### Curl command (JSON)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/json'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Discussions('smlWeEw5ZxLDgOIGsGJP4o')/Comments"
-d@C:/payloads/payload.json
```

### Payload file (JSON)

```
{
  "Text": "hello @@m{0} and @@m{1}",
  "AtMentions": [
    { "__metadata": { "uri": "Members('4PcGbb1FT5zrklWDDAvxRr')"} },
    { "__metadata": { "uri": "Members('OydLY2ajYmbQ9sBImlRhpQ')"} }
  ]
}
```

### JSON Response

```
{ "d": { "results": {
  " __metadata": {
    "uri": "Comments('MWdtYwBzYhI9dcXFqCjmE1')",
    "type": "SAPJam.Comment"
  },
  "Id": "MWdtYwBzYhI9dcXFqCjmE1",
  "CreatedAt": "/Date(1415173646000)/",
  "LastModifiedAt": "/Date(1415173646000)/",
  "Text": "hello @user1235 and @user",
  "TextWithPlaceholders": "hello @@m{0} and @@m{1}",
  "Liked": false,
  "LikesCount": 0,
  "CanDelete": true,
  /* [Navigation Properties have been removed from here.] */
}}}
}}
```

### Curl command (XML)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Discussions('smlWeEw5ZxLDgOIGsGJP4o')/Comments"
-d@C:/payloads/payload.xml
```

### Payload file (XML)

```
<entry>
  <content>
    <properties>
      <Text>hello @@m{0} and @@m{1}</Text>
    </properties>
  </content>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/AtMentions"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('4PcGbb1FT5zrklWDDAvxRr')"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/AtMentions"
```

```

      href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('OydLY2ajYmbQ9sBImlRhpQ')"/>
</entry>

```

## XML Response

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
Comments('8owrxslaph5UzXpOXOulG7')</id>
  <title type="text">Comments('8owrxslaph5UzXpOXOulG7')</title>
  <updated>2014-11-05T07:45:56+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="Comments('8owrxslaph5UzXpOXOulG7') "
href="Comments('8owrxslaph5UzXpOXOulG7')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.Comment" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">8owrxslaph5UzXpOXOulG7</d:Id>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2014-11-05T07:45:55Z</
d:CreatedAt>
      <d:LastModifiedAt m:type="Edm.DateTimeOffset">2014-11-05T07:45:55Z</
d:LastModifiedAt>
      <d:Text m:type="Edm.String">hello @user1235 and @user</d:Text>
      <d:TextWithPlaceholders m:type="Edm.String">hello @@m{0} and @@m{1}</
d:TextWithPlaceholders>
      <d:Liked m:type="Edm.Boolean">>false</d:Liked>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:CanDelete m:type="Edm.Boolean">>true</d:CanDelete>
    </m:properties>
  </content>
</entry>

```

## [GET] /WallComments('{key}')

### Curl command (JSON)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
  -H 'Content-Type: application/json' -H 'Accept: application/json'
  "https://{jam#} [page 246].sapjam.com/api/v1/OData/
WallComments('i9Qi7jP4V6ovrIkkyG9uQ2') "

```

### JSON Response

```

{"d": {"results": {
  "metadata": {
    "uri": "WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')",
    "type": "SAPJam.WallComment"
  },
  "Id": "i9Qi7jP4V6ovrIkkyG9uQ2",
  "Comment": "hello @test144 and @admin ",
  "CommentWithPlaceholders": "hello @@m{0} and @@m{1} ",
  "Liked": false,
  "LikesCount": 0,

```

```

    "ReplyCount": 0,
    "CreatedAt": "/Date(1415174235000)/",
    "LastModifiedAt": "/Date(1415174235000)/",
    "CanDelete": true,
    /* [Navigation Properties have been removed from here.] */
  }}}

```

### Curl command (XML)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/
WallComments('i9Qi7jP4V6ovrIkkyG9uQ2') "

```

### XML Response

```

<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')</id>
  <title type="text">WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')</title>
  <updated>2014-11-05T07:58:17+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="WallComments('i9Qi7jP4V6ovrIkkyG9uQ2') "
    href="WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.WallComment" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">i9Qi7jP4V6ovrIkkyG9uQ2</d:Id>
      <d:Comment m:type="Edm.String">hello @test144 and @admin </d:Comment>
      <d:CommentWithPlaceholders m:type="Edm.String">hello @@m{0} and
@@m{1} </d:CommentWithPlaceholders>
      <d:Liked m:type="Edm.Boolean">false</d:Liked>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:ReplyCount m:type="Edm.Int32">0</d:ReplyCount>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2014-11-05T07:57:15Z</
d:CreatedAt>
      <d:LastModifiedAt m:type="Edm.DateTimeOffset">2014-11-05T07:57:15Z</
d:LastModifiedAt>
      <d:CanDelete m:type="Edm.Boolean">true</d:CanDelete>
    </m:properties>
  </content>
</entry>

```

## [GET] /WallComments('{key}')/AtMentions

### Curl command (JSON)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/json'
"https://{jam#} [page 246].sapjam.com/api/v1/OData/
WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions"

```

## JSON Response

```
{
  "d": {
    "results": [
      {
        "__metadata": {
          "uri": "Members('6q2CXkDtrnUZPksZIiDh9n')",
          "type": "SAPJam.Member"
        },
        "Id": "6q2CXkDtrnUZPksZIiDh9n",
        "FirstName": null,
        "LastName": null,
        "Nickname": null,
        "Title": null,
        "Email": "test144@test.com",
        "FullName": "test144",
        "Role": "company",
        "IsFollowing": false,
        /* [Navigation Properties have been removed from here.] */
      },
      /* [Multiple AtMentions entries would appear here.] */
    ]
  }
}
```

## Curl command (XML)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
'https://{jam#} [page 246].sapjam.com/api/v1/OData/WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions'
```

## XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions</id>
  <link rel="self" title="WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/WallComments('i9Qi7jP4V6ovrIkkyG9uQ2')/AtMentions"/>
  <updated>2014-11-05T07:59:10+00:00</updated>
  <entry>
    <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/Members('6q2CXkDtrnUZPksZIiDh9n')</id>
    <title type="text">Members('6q2CXkDtrnUZPksZIiDh9n')</title>
    <updated>2014-11-05T07:59:10+00:00</updated>
    <author>
      <name/>
    </author>
    <link rel="edit" title="Members('6q2CXkDtrnUZPksZIiDh9n') "
      href="Members('6q2CXkDtrnUZPksZIiDh9n')"/>
    <!--[Navigation Properties have been removed from here.]-->
    <category term="SAPJam.Member" scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
    <content type="application/atom+xml">
      <m:properties>
        <d:Id m:type="Edm.String">6q2CXkDtrnUZPksZIiDh9n</d:Id>
        <d:FirstName m:type="Edm.String" m:null="true"/>
        <d:LastName m:type="Edm.String" m:null="true"/>
        <d:Nickname m:type="Edm.String" m:null="true"/>
        <d:Title m:type="Edm.String" m:null="true"/>
        <d:Email m:type="Edm.String">test144@test.com</d:Email>
        <d:FullName m:type="Edm.String">test144</d:FullName>
      </m:properties>
    </content>
  </entry>
</feed>
```

```

        <d:Role m:type="Edm.String">company</d:Role>
        <d:IsFollowing m:type="Edm.Boolean">false</d:IsFollowing>
    </m:properties>
</content>
</entry>
<!--[Multiple AtMentions entries would appear here.]-->
</feed>

```

## [POST] /Ideas('{key}')/Posts

### Curl command (JSON)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
-H 'Content-Type: application/json' -H 'Accept: application/json'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Ideas('ygl1nzb7lEyBRT6o0FDNVva')/Posts"
-d@C:/payloads/payload.json

```

### Payload file (JSON)

```

{
  "Comment": "hello @@m{0} and @@m{1}",
  "AtMentions": [
    { "__metadata": { "uri": "Members('4PcGbblFT5zrklWDDAvxRr')"} },
    { "__metadata": { "uri": "Members('OydLY2ajYmbQ9sBIm1RhpQ')"} }
  ]
}
EOF

```

### JSON Response

```

{"d": {"results": {
  "__metadata": {
    "uri": "WallComments('ODLrKxIxrE9V9DyHyHYQsR')",
    "type": "SAPJam.WallComment"
  },
  "Id": "ODLrKxIxrE9V9DyHyHYQsR",
  "Comment": "hello @user1235 and @user",
  "CommentWithPlaceholders": "hello @@m{0} and @@m{1}",
  "Liked": false,
  "LikesCount": 0,
  "ReplyCount": 0,
  "CreatedAt": "/Date(1415174503000)/",
  "LastModifiedAt": "/Date(1415174503000)/",
  "CanDelete": true,
  /* [Navigation Properties have been removed from here.] */
}}}

```

### Curl command (XML)

```

curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed
from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Ideas('ygl1nzb7lEyBRT6o0FDNVva')/Posts"
-d@C:/payloads/payload.xml

```



## Payload file (XML)

```
<entry>
  <content>
    <properties>
      <Comment>hello @@m{0} and @@m{1}</Comment>
    </properties>
  </content>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AtMentions"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('4PcGbb1FT5zrklWDDAvxRr')"/>
  <link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
AtMentions"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
Members('OydLY2ajYmbQ9sBIm1RhpQ')"/>
</entry>
EOF
```

## XML Response

```
<?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">
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
WallComments('PrsBrTdzyG3rbGRGPFOOV1')</id>
  <title type="text">WallComments('PrsBrTdzyG3rbGRGPFOOV1')</title>
  <updated>2014-11-05T08:01:26+00:00</updated>
  <author>
    <name/>
  </author>
  <link rel="edit" title="WallComments('PrsBrTdzyG3rbGRGPFOOV1') "
    href="WallComments('PrsBrTdzyG3rbGRGPFOOV1')"/>
  <!--[Navigation Properties have been removed from here.]-->
  <category term="SAPJam.WallComment" scheme="http://schemas.microsoft.com/ado/
2007/08/dataservices/scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String">PrsBrTdzyG3rbGRGPFOOV1</d:Id>
      <d:Comment m:type="Edm.String">hello @user1235 and @user</d:Comment>
      <d:CommentWithPlaceholders m:type="Edm.String">hello @@m{0} and
@@m{1}</d:CommentWithPlaceholders>
      <d:Liked m:type="Edm.Boolean">false</d:Liked>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:ReplyCount m:type="Edm.Int32">0</d:ReplyCount>
      <d:CreatedAt m:type="Edm.DateTimeOffset">2014-11-05T08:01:26Z</
d:CreatedAt>
      <d:LastModifiedAt m:type="Edm.DateTimeOffset">2014-11-05T08:01:26Z</
d:LastModifiedAt>
      <d:CanDelete m:type="Edm.Boolean">true</d:CanDelete>
    </m:properties>
  </content>
</entry>
```

## Bad Requests

This applies to all Placeholder/AtMention endpoints:

1. Mismatch between placeholder index and link position (starts at 0)
2. Mismatch between number of placeholders and number of links

## [POST] /Discussions('{key}')/Comments (1)

### Curl command (XML)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Content-Type: application/atom+xml' -H 'Accept: application/atom+xml'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Discussions('smlWeEw5ZxLDgOIGsGJP4o')/Comments"
-d@C:/payloads/payload.xml
```

### Payload file (XML)

```
<entry>
  <content>
    <properties>
      <Text>hello @@m{0} and @@m{2}</Text>
    </properties>
  </content>
  <!--[Navigation Properties have been removed from here.]-->
</entry>
```

### XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<error xmlns="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
  <code/>
  <message xml:lang="en">No @mention link found for placeholder: @@m{2}</
message>
</error>
```

## [POST] /Discussions('{key}')/Comments (2)

### Curl command (XML)

```
curl -i -H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-H 'Accept: application/atom+xml' -H 'Content-Type: application/atom+xml'
-X POST "https://{jam#} [page 246].sapjam.com/api/v1/OData/
Discussions('smlWeEw5ZxLDgOIGsGJP4o')/Comments"
-d@C:/payloads/payload.xml
```

### Payload file (XML)

```
<entry>
  <content>
    <properties>
      <Text>hello @@m{0} and @@m{1}</Text>
    </properties>
  </content>
  <!--[Navigation Properties have been removed from here.]-->
</entry>
```

### XML Response

```
<?xml version="1.0" encoding="UTF-8"?>
<error xmlns="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
```

```
<code/>
<message xml:lang="en">No @mention link found for placeholder: @@m{1}</
message>
</error>
```

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to set an [@mention placeholder \[page 364\]](#)?
- Do you understand the example, [\[GET\] /Comments\('{key}'\) \[page 365\]](#)?
- Do you understand the example, [\[GET\] /Comments\('{key}'\)/AtMentions?](#)
- Do you understand the example, [\[POST\] /Discussions\('{key}'\)/Comments?](#)
- Do you understand the example, [\[GET\] /WallComments\('{key}'\) \[page 369\]](#)?
- Do you understand the example, [\[GET\] /WallComments\('{key}'\)/AtMentions \[page 370\]](#)?
- Do you understand the example, [\[POST\] /Ideas\('{key}'\)/Posts \[page 372\]](#)?
- Do you understand the problem in the bad request example, [\[POST\] /Discussions\('{key}'\)/Comments \(1\) \[page 374\]](#)?
- Do you understand the problem in the second bad request example, [\[POST\] /Discussions\('{key}'\)/Comments \(2\) \[page 374\]](#)?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [The Search and related API \[page 375\]](#).

## 8.1.2.8 The Search and related API

The Search and SearchSummary entities provide a search capability that potentially covers all material within your organization's SAP Jam content.

## Learning Objectives

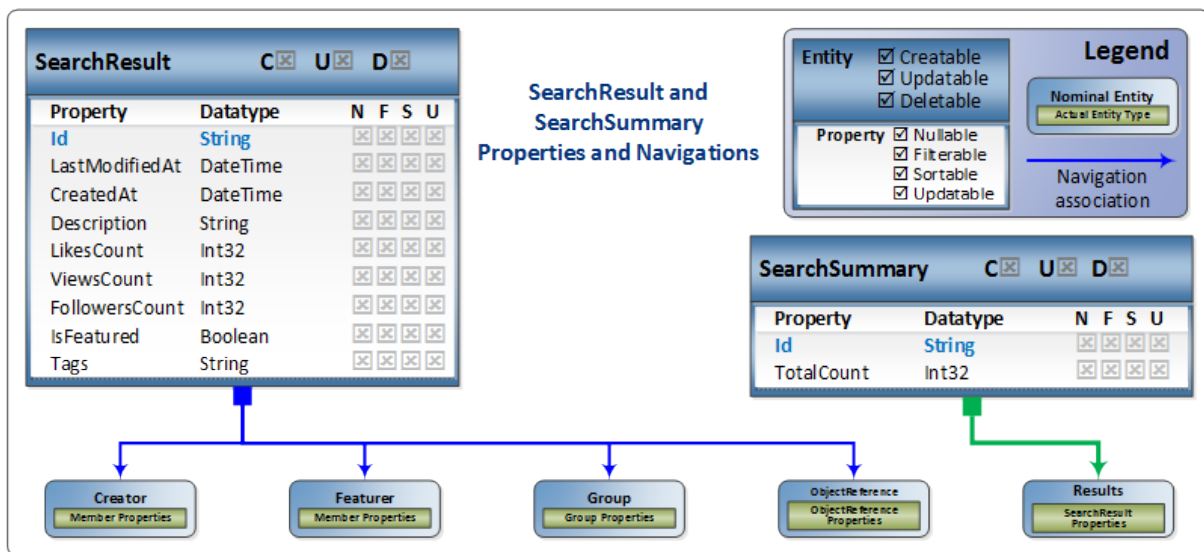
The learning objectives for this page are:

- Know what information is required to form a [request to search all of your organization's instance of SAP Jam \[page 376\]](#).
- Know what the [available endpoints \[page 377\]](#) are for performing searches by using the API.
- Know how to form a [basic search call \[page 377\]](#) using the API.
- Know the [possible search parameters \[page 378\]](#).
- Know how to [save a search \[page 379\]](#).
- Know how to [expand search results \[page 379\]](#).

## Properties and Navigations

The following diagram and the itemized list that follows it are provided to give you, the new SAP Jam OData API user, a good sense of what data (or "properties") the depicted entities represent. The diagrams also show the navigations from each of the entities covered. Navigations can be thought of as URL extensions from the base entity to closely related entities. For example, in the URL `[GET] /SearchSummaries('{id}')/Results`, "Results" is the navigation. The API call in this example retrieves the SearchResults associated with the specified SearchSummary.

There are two OData entities devoted to SAP Jam searches, as shown in the list and the diagram below.



The SearchResult and SearchSummary entities' properties and navigations

1. **SearchResult:** provides information on the results of a specified search term. SearchResult entity properties are:
  - **Id:** The [unique ID \[page 246\]](#) of the SearchResult.
  - **LastModifiedAt:** The date and time that the item that the SearchResult refers to was last modified.
  - **CreatedAt:** The date and time that the item that the SearchResult refers to was created.
  - **Description:** The description of the item that the SearchResult refers to.
  - **LikesCount:** The total number of likes that the item that the SearchResult refers to has received.
  - **ViewsCount:** The total number of times that members have viewed the item that the SearchResult refers to.
  - **FollowersCount:** The total number of followers that the item that the SearchResult refers to has.
  - **IsFeatured:** Whether the item that the SearchResult refers to is featured.
  - **Tags:** A comma-separated list of tags that have been applied to the item that the SearchResult refers to.
2. **SearchSummary:** provides information on the number of search results for a specific query. SearchSummary entity properties are:
  - **Id:** The [unique ID \[page 246\]](#) of the SearchSummary.
  - **TotalCount:** A count of the matching search results.

## Available Endpoints

The following list of API calls is shown only to give you, the new SAP Jam OData API user, a good sense of what operations are available for the SearchResult and SearchSummary entities. The available endpoints are:

- [\[GET\] /Search](#) — Retrieves a collection of SearchResult entries based on the search parameters specified in the URL.
- [\[GET\] /SearchResults\('{id}'\)](#) — Retrieves the specified SearchResult.
- [\[GET\] /SearchResults\('{id}'\)/Creator](#) — Retrieves the Member information on the Creator of the specified SearchResult.
- [\[GET\] /SearchResults\('{id}'\)/Featurer](#) — Retrieves the Member information on the Featurer of the specified SearchResult. This will only return results if the specified SearchResult has been featured.
- [\[GET\] /SearchResults\('{id}'\)/Group](#) — Retrieves the information on the Group that contains the specified SearchResult.
- [\[GET\] /SearchResults\('{id}'\)/ObjectReference](#) — Retrieves the information required to access the SearchResults that are specified by the SearchResults "key" or "Id".
- [\[GET\] /SearchSummary](#) — Retrieves a count of the matches for the specified search parameters. This operation also returns an Id, which is an encoded string of the search parameters.
- [\[GET\] /SearchSummaries\('{id}'\)](#) — Retrieves a count of the matches for the specified search parameters. This operation also returns an Id, which is an encoded string of the search parameters.
- [\[GET\] /SearchSummaries\('{id}'\)/Results](#) — Retrieves a collection of all search result entries for search parameters specified in the encoded SearchSummaries key.

## Basic search call

At its most basic, you can simply set a `Query` term for your search, as is shown in the following example (shown in XML and JSON).

### Curl Command (XML):

```
curl -i -H "Authorization: OAuth <!--[The OAuth1.0a protocol parameters have been removed from here.]-->"  
      -H "Accept: application/atom+xml"  
      "https://{jam#} [page 246].sapjam.com/api/v1/OData/SearchSummary?  
Query='{search_term}'"
```

### XML Response:

```
<?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">  
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/  
SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')</id>  
  <title type="text">SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')</title>  
  <updated>2015-06-16T20:36:55+00:00</updated>  
  <author>  
    <name/>  
  </author>  
  <link rel="edit" title="SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0') "  
    href="SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')"/>
```

```
<link rel="http://schemas.microsoft.com/ado/2007/08/dataservices/related/
Results"
      type="application/atom+xml;type=feed" title="Results"
      href="SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')/Results"/>
<category term="SAPJam.SearchSummary"
      scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/scheme"/>
<content type="application/atom+xml">
  <m:properties>
    <d:Id m:type="Edm.String">eyJRdWVyeSI6ImxlyWRzIn0</d:Id>
    <d:TotalCount m:type="Edm.Int32">106</d:TotalCount>
  </m:properties>
</content>
</entry>
```

### Curl Command (JSON):

```
curl -i -H "Authorization: Bearer /* [An OAuth2.0 access token has been removed
from here.] */"
      -H "Accept: application/json"
      "https://{jam#} [page 246].sapjam.com/api/v1/odata/SearchSummary?
Query='{search_term}'"
```

### JSON Response:

```
{ "d": { "results": {
  "___metadata": {
    "uri": "SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')",
    "type": "SAPJam.SearchSummary"
  },
  "Id": "eyJRdWVyeSI6ImxlyWRzIn0",
  "TotalCount": 106,
  "Results": { "___deferred": { "uri":
    "SearchSummaries('eyJRdWVyeSI6ImxlyWRzIn0')/Results" } }
  } } }
```

## Possible search parameters

In addition to the `Query` term, you can set five parameters to narrow your search.

Parameter	Description
Query	The term that you are searching for.
Category	The type of item that you are searching for. The valid Category options are: people, documents, content, photos, pages, blog_entries, groups, links, comments, member_statuses, forums, and polls.
Tag	The tag applied to the item that you are searching for.
Creator	The <a href="#">unique ID [page 246]</a> of the Member who created the item that you are searching for.
Group	The <a href="#">unique ID [page 246]</a> of the Group in which the item that you are searching for is located.

## Curl Command (XML):

The search parameters are illustrated in the following curl command.

```
curl -i -H "Authorization: OAuth <!--[The OAuth1.0a protocol parameters have been removed from here.]-->"
-H "Accept: application/atom+xml"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/SearchSummary?
Query='{search_term}'

&Category='documents'&Tag='leads'&Creator='oM1lKYSJyHHbBkfDYaO3KP'&Group='7pMvTbn
ZK66XYCuuRuKl6W'"
```

## Saving Searches

Once you have run a search, you can "save" it, by recording or posting the Search URL.

What is actually done here, is that the search query is set as an encoded string that is shown in the URL of the returned "SearchSummaries" result, and following this URL will display the results of the search.

## Expanding Search results

You can expand the results of your search to show details of any of its navigations. For example, the following search would return the expanded result of the ObjectReference:

```
curl -i -H "Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->"
-H "Accept: application/json"
"https://{jam#} [page 246].sapjam.com/api/v1/OData/
SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/Results?\
$expand=ObjectReference"
```

The preceding curl command will produce something like the following result.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xml:base="https://{jam#} [page 246].sapjam.com/api/v1/OData" 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>SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/Results</title>
  <id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
    SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/Results?%5C
%24expand=ObjectReference</id>
  <link rel="self" title="SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/Results"
    href="https://{jam#} [page 246].sapjam.com/api/v1/OData/
    SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/Results?%5C
%24expand=ObjectReference"/>
  <link rel="next"
    title="SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/
    Results?%24skiptoken=<!--[Skip token removed.]-->%5C
%24expand=ObjectReference"
    href="SearchSummaries('eyJRdWVyeSI6Imx1YWRzIn0')/
    Results?%24skiptoken=<!--[Skip token removed.]-->%5C
%24expand=ObjectReference"/>
  <updated>2015-06-16T20:48:05+00:00</updated>
  <entry>
```

```

<id>https://{jam#} [page 246].sapjam.com/api/v1/OData/
  SearchResults('<!--[Result key removed.]-->')</id>
<title type="text"
  >SearchResults('<!--[Result key removed.]-->')</title>
<updated>2015-06-16T20:48:05+00:00</updated>
<author>
  <name/>
</author>
<link rel="edit"
  title="SearchResults('<!--[Result key removed.]-->') "
  href="SearchResults('<!--[Result key removed.]-->')"/>
<!--[Navigation Properties have been removed from here.]-->
<category term="SAPJam.SearchResult"
  scheme="http://schemas.microsoft.com/ado/2007/08/dataservices/
scheme"/>
  <content type="application/atom+xml">
    <m:properties>
      <d:Id m:type="Edm.String"
        ><!--[Result key removed.]--></d:Id>
      <d:LastModifiedAt
m:type="Edm.DateTimeOffset">2014-06-02T06:09:49Z</d:LastModifiedAt>
      <d:CreatedAt m:type="Edm.DateTimeOffset" m:null="true"/>
      <d:Description m:type="Edm.String">Leads, GM4, 02.06.; 08:09
Click here to see the
      details.</d:Description>
      <d:LikesCount m:type="Edm.Int32">0</d:LikesCount>
      <d:ViewsCount m:type="Edm.Int32">0</d:ViewsCount>
      <d:FollowersCount m:type="Edm.Int32">0</d:FollowersCount>
      <d:IsFeatured m:type="Edm.Boolean">>false</d:IsFeatured>
      <d:Tags m:type="Edm.String"/>
    </m:properties>
  </content>
</entry>
<!--[Multiple SearchResults entries would appear here.]-->
</feed>

```

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is required to form [a request to search all of your organization's instance of SAP Jam \[page 376\]](#)?
- Do you know what the [available endpoints \[page 377\]](#) are for performing searches by using the API?
- Do you know how to form a [basic search call \[page 377\]](#) using the API?
- Do you know the [possible search parameters \[page 378\]](#)?
- Do you know how to [save a search \[page 379\]](#)?
- Do you know how to [expand search results \[page 379\]](#)?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Understanding OData metadata \[page 381\]](#).



## 8.1.2.9 Understanding OData metadata

The OData \$metadata file is a CSDL file that is made available to clients to help them discover the structure and organization of the entities, navigations, and the service operations that are available to manage resources beyond the usual create, retrieve, update, or delete operations.

The SAP Jam Collaboration OData API \$metadata file is located in the service root directory; for example:

```
https://{jam#} [page 246].sapjam.com/api/v1/OData/$metadata
```

### i Note

The {jam#} can be easily found by looking at the URL of any SAP Jam Collaboration page in your organization's instance of SAP Jam.

## Learning Objectives

The learning objectives for this page are to understand the content of each of the five major sections of the \$metadata file:

- Know what information is provided in [the EntityType section \[page 381\]](#).
- Know what information is provided in [the Association section \[page 383\]](#).
- Know what information is provided in [the EntitySet section \[page 383\]](#).
- Know what information is provided in [the AssociationSet section \[page 385\]](#).
- Know what information is provided in [the FunctionImport \(Service Operations\) section \[page 386\]](#).

## The EntityType section

Each EntityType is abstract data model of a type of resource, a generalized collection of the properties that describe each type of entity. An EntityType is roughly equivalent to a database table. For example:

```
<EntityType Name="ContentItem" m:HasStream="true">
  <Key>
    <PropertyRef Name="Id"/>
    <PropertyRef Name="ContentItemType"/>
  </Key>
  <Property Name="Id" Type="Edm.String" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
  <Property Name="Name" Type="Edm.String" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="true"/>
  <Property Name="ContentItemType" Type="Edm.String" Nullable="false"
sap:filterable="true" sap:sortable="false" sap:updatable="false"/>
  <Property Name="Description" Type="Edm.String" Nullable="true"
sap:filterable="false" sap:sortable="false" sap:updatable="true"/>
  <Property Name="CreatedAt" Type="Edm.DateTimeOffset" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
  <Property Name="LastModifiedAt" Type="Edm.DateTimeOffset" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
  <Property Name="ViewsCount" Type="Edm.Int32" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
```

```

    <Property Name="Liked" Type="Edm.Boolean" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="true"/>
    <Property Name="LikesCount" Type="Edm.Int32" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="IsFeatured" Type="Edm.Boolean" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="PermissionType" Type="Edm.String" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="true"/>
    <Property Name="DocumentSize" Type="Edm.Int32" Nullable="true"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="FileName" Type="Edm.String" Nullable="true"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="ConversionStatus" Type="Edm.String" Nullable="true"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="IsCheckedOut" Type="Edm.Boolean" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="true"/>
    <Property Name="FeedCommentsCount" Type="Edm.Int32" Nullable="false"
sap:filterable="false" sap:sortable="false" sap:updatable="false"/>
    <Property Name="ExternallyCreatedAt" Type="Edm.DateTimeOffset"
Nullable="false" sap:filterable="false" sap:sortable="false"
sap:updatable="false"/>
    <NavigationProperty Name="CheckedOutByMembers"
Relationship="SAPJam.ContentItem_CheckedOutByMembers"
FromRole="ContentItem_Source" ToRole="Member_Target"/>
    <NavigationProperty Name="Creator" Relationship="SAPJam.ContentItem_Creator"
FromRole="ContentItem_Source" ToRole="Member_Target"/>
    <NavigationProperty Name="FeedEntries"
Relationship="SAPJam.ContentItem_FeedEntries" FromRole="ContentItem_Source"
ToRole="FeedEntry_Target"/>
    <NavigationProperty Name="Group" Relationship="SAPJam.ContentItem_Group"
FromRole="ContentItem_Source" ToRole="Group_Target"/>
    <NavigationProperty Name="LastModifier"
Relationship="SAPJam.ContentItem_LastModifier" FromRole="ContentItem_Source"
ToRole="Member_Target"/>
    <NavigationProperty Name="Likers" Relationship="SAPJam.ContentItem_Likers"
FromRole="ContentItem_Source" ToRole="Member_Target"/>
    <NavigationProperty Name="ParentFolder"
Relationship="SAPJam.ContentItem_ParentFolder" FromRole="ContentItem_Source"
ToRole="Folder_Target"/>
</EntityType>

```

This section lists the following information for each entity:

- **Name**—The name of the entity, such as Comment, Group, and Member.
- **HasStream**—Set only for entities that represent BLOBs (Binary Large Objects).
- **Key**—The name or names of one or more properties that are required to uniquely identify a specific resource.
- **Properties**—The various bits of data that are available for each instance of the entity; roughly equivalent to a field in a database table.
  - The `Type` attribute indicates the data type of the property.
  - The `Nullable` attribute indicates whether the property can have a null value (not be given a value).
  - The `sap:filterable` attribute indicates whether the entity's feed results can be filtered (using [\\$filter](#) [page 327]).
  - The `sap:sortable` attribute indicates whether the entity's feed results can be sorted (using [\\$orderby](#) [page 329]).
  - The `sap:updatable` attribute indicates whether the entity can be updated (using a `PATCH` operation).
- **Navigation Properties**—The secondary entities, related to this entity, that can be examined for further information relevant to a specific instance of the primary entity.
  - The `Name` attribute identifies the navigation property.

- The `Relationship` attribute refers to the name of the Association.
- The `FromRole` and `ToRole` indicate the direction of the navigation.

## The Association section

Defines a peer-to-peer relationship between participating entity types. An Association contains two ends, or "roles" (role is the proper term specified in the Conceptual Schema Definition Language [CSDL] specification). You can define multiplicities at both ends.

For example, the following Association is specified in the SAP Jam \$metadata page:

```
<Association Name="Group_Creator">
  <End Role="Group_Source" Type="SAPJam.Group" Multiplicity="0..1"/>
  <End Role="Member_Target" Type="SAPJam.Member" Multiplicity="0..1"/>
</Association>
```

This section lists the following information for each association:

- **Name**—The name of the association, which is a two-part name that shows the two ends of the association.
- **End**—Each end of the association, which includes the following information:
  - **Role**—The entity type of the Source and Target ends of the association.
  - **Type**—The container and entity name of the associated entities of each end of the association.
  - **Multiplicity**—The number of entities that can be involved. The possible values are either "0..1" or "\*".

## The EntitySet section

A list of the full set of EntityTypes available in the SAP Jam OData service, for example: This section lists the "connections" between a primary entity and its navigation entities. Each association describes the "Source", the "Target", and the "Multiplicity" (the number if instances) that is appropriate for each the target and the source.

```
<EntityContainer Name="SAPJam" m:IsDefaultEntityContainer="true">
  <EntitySet Name="Activities" EntityType="SAPJam.Activity"
    sap:creatable="true" sap:updatable="false" sap:deletable="false"/>
  <EntitySet Name="AwayAlerts" EntityType="SAPJam.AwayAlert"
    sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
  <EntitySet Name="Comments" EntityType="SAPJam.Comment" sap:creatable="true"
    sap:updatable="true" sap:deletable="true"/>
  <EntitySet Name="Companies" EntityType="SAPJam.Company"
    sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
  <EntitySet Name="ContentItems" EntityType="SAPJam.ContentItem"
    sap:creatable="true" sap:updatable="true" sap:deletable="true"/>
  <EntitySet Name="ContentListItems" EntityType="SAPJam.ContentListItem"
    sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
  <EntitySet Name="CustomGroupTemplates"
    EntityType="SAPJam.CustomGroupTemplate" sap:creatable="false"
    sap:updatable="true" sap:deletable="false"/>
  <EntitySet Name="Discussions" EntityType="SAPJam.Discussion"
    sap:creatable="true" sap:updatable="true" sap:deletable="true"/>
  <EntitySet Name="Events" EntityType="SAPJam.Event" sap:creatable="true"
    sap:updatable="true" sap:deletable="true"/>
```

```

    <EntitySet Name="EventResponses" EntityType="SAPJam.EventResponse"
sap:creatable="true" sap:updatable="true" sap:deletable="false"/>
    <EntitySet Name="ExternalApplications"
EntityType="SAPJam.ExternalApplication" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="ExternalObjects" EntityType="SAPJam.ExternalObject"
sap:creatable="true" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="ExternalObjectTypes" EntityType="SAPJam.ExternalObjectType"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="FeedEntries" EntityType="SAPJam.FeedEntry"
sap:creatable="true" sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="FeedEntryImages" EntityType="SAPJam.FeedEntryImage"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="FeedFilters" EntityType="SAPJam.FeedFilter"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Folders" EntityType="SAPJam.Folder" sap:creatable="true"
sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="Forums" EntityType="SAPJam.Forum" sap:creatable="true"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="ForumItems" EntityType="SAPJam.ForumItem"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Groups" EntityType="SAPJam.Group" sap:creatable="true"
sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="GroupExternalObjects"
EntityType="SAPJam.GroupExternalObject" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="GroupGadgets" EntityType="SAPJam.GroupGadget"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="GroupGadgetObjects" EntityType="SAPJam.GroupGadgetObject"
sap:creatable="true" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="GroupMemberships" EntityType="SAPJam.GroupMembership"
sap:creatable="false" sap:updatable="false" sap:deletable="true"/>
    <EntitySet Name="GroupTemplates" EntityType="SAPJam.GroupTemplate"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Ideas" EntityType="SAPJam.Idea" sap:creatable="true"
sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="Images" EntityType="SAPJam.Image" sap:creatable="true"
sap:updatable="false" sap:deletable="true"/>
    <EntitySet Name="Kudos" EntityType="SAPJam.Kudo" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Members" EntityType="SAPJam.Member" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="MemberKudos" EntityType="SAPJam.MemberKudo"
sap:creatable="true" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="MemberProfiles" EntityType="SAPJam.MemberProfile"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Notifications" EntityType="SAPJam.Notification"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="ObjectReferences" EntityType="SAPJam.ObjectReference"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="PhoneNumbers" EntityType="SAPJam.PhoneNumber"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Questions" EntityType="SAPJam.Question"
sap:creatable="true" sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="SearchResults" EntityType="SAPJam.SearchResult"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="SearchSummaries" EntityType="SAPJam.SearchSummary"
sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="SecondaryEmailAddresses"
EntityType="SAPJam.SecondaryEmailAddress" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="SystemGroupTemplates"
EntityType="SAPJam.SystemGroupTemplate" sap:creatable="false"
sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="Tasks" EntityType="SAPJam.Task" sap:creatable="true"
sap:updatable="true" sap:deletable="true"/>
    <EntitySet Name="TaskAssignments" EntityType="SAPJam.TaskAssignment"
sap:creatable="true" sap:updatable="true" sap:deletable="true"/>

```

```

    <EntitySet Name="ThumbnailImages" EntityType="SAPJam.ThumbnailImage"
    sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="ThumbnailKudoImages" EntityType="SAPJam.ThumbnailKudoImage"
    sap:creatable="false" sap:updatable="false" sap:deletable="false"/>
    <EntitySet Name="WallComments" EntityType="SAPJam.WallComment"
    sap:creatable="true" sap:updatable="true" sap:deletable="true"/>

```

This section lists the following information for each entity set:

- **Name**—The name of the EntitySet.
- **EntityType**—This section lists the "connections" between a primary entity and its navigation—The container and entity name of the entity type.
- **sap:creatable**—Whether the entity can be created (using a `POST` operation). Note that many entities can only be created through navigation operations from their parent entity, such as `Group`.
- **sap:updatable**—Whether the entity can be updated (using a `PATCH` operation).
- **sap:deletable**—Whether the entity can be deleted (using a `DELETE` operation).

## The AssociationSet section

While the OData Association specifies the entity type at both ends of the relationship, the OData AssociationSet specifies the entity sets that contain instances of the specified Association. Relationship instances exist between entities belonging to these entity sets. Note that the AssociationSet's `Association` attribute points to the name of the Association.

For example, the following AssociationSet is specified in the SAP Jam \$metadata page:

```

<AssociationSet Name="Group_ContentItems"
Association="SAPJam.Group_ContentItems" sap:creatable="true"
sap:deletable="true" sap:updatable="true">
    <End EntitySet="Groups" Role="Group_Source"/>
    <End EntitySet="ContentItems" Role="ContentItem_Target"/>
</AssociationSet>

```

This section lists the following information for each association:

- **Name**—The name of the association set, which is a two-part name that shows the two ends of the association.
- **Association**—The name of the container and the association, which is a two-part name that shows the two ends of the association.
- **sap:creatable**—Whether the entity can be created using this AssociationSet (in a `POST` operation). Note that many entities can only be created through navigation operations from their parent entity, such as `Group`.
- **sap:updatable**—Whether the entity can be updated using this AssociationSet (in a `PATCH` operation).
- **sap:deletable**—Whether the entity can be deleted using this AssociationSet (in a `DELETE` operation).
- **End**—Each end of the association, which includes the following information:
  - **EntitySet**—The entity set for each end.
  - **Role**—a two-part name that shows the entity type of the Source and Target ends of the association set.

## The FunctionImport (Service Operations) section

This section lists the SAP Jam Collaboration OData API's Service Operations, which are custom behaviors that do not map to the uniform interface of the OData service.

```
<FunctionImport Name="ContentListItems_CopyMultiple" m:HttpMethod="POST">
  <Parameter Name="TargetGroupId" Type="Edm.String"/>
  <Parameter Name="TargetFolderId" Type="Edm.String"/>
  <Parameter Name="Ids" Type="Edm.String"/>
</FunctionImport>
<FunctionImport Name="FeedEntries_Replies" m:HttpMethod="GET"
  EntitySet="FeedEntries" ReturnType="Collection(SAPJam.FeedEntry)"/>
<FunctionImport Name="GroupMembership_RemoveFromGroup" m:HttpMethod="POST">
  <Parameter Name="GroupId" Type="Edm.String"/>
  <Parameter Name="MemberId" Type="Edm.String"/>
</FunctionImport>
```

This section lists the following information for each association:

- **Name**—The name of the association set, which is a two-part name that shows the two ends of the association.
- **m:HttpMethod**—The HTTP request type verb. The valid options are POST or GET.
- **EntitySet**—The entity set that the operation will work on.
- **ReturnType**—Either the container and entity type, or a collection of the container and entity type, that will be returned.
- **Parameter**—The definition of each parameter of the function (service operation):
  - **Name**—The name of the parameter.
  - **Type**—Edm and the data type of the parameter.

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know what information is provided in [the EntityType section \[page 381\]](#)?
- Do you know what information is provided in [the Association section \[page 383\]](#)?
- Do you know what information is provided in [the EntitySet section \[page 383\]](#)?
- Do you know what information is provided in [the AssociationSet section \[page 385\]](#)?
- Do you know what information is provided in [the FunctionImport \(Service Operations\) section \[page 386\]](#)?

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Batching API calls together \[page 386\]](#).

### 8.1.2.10 Batching API calls together

Batch requests enable members to bundle OData API requests into one consolidated request and get back a consolidated response from the request. SAP Jam Collaboration supports \$batch requests for GET OData

requests. Currently, changesets are not supported as part of the batch request. A maximum of ten requests can be used as part of a single batch request.

## Learning Objectives

The learning objectives for this page are:

- Know how to form [\\$batch requests \[page 387\]](#).
- Understand the [request example \[page 387\]](#).
- Know what to expect in [\\$batch responses \[page 388\]](#).
- Know how to recognize a [request with an error \[page 388\]](#).
- Know what to expect in a [response to a request with an error \[page 388\]](#).

## `$batch` Requests

To make a `$batch` request, you must:

- Make a POST request to `https://{jam#} [page 246].sapjam.com/api/v1/OData/$batch`.
- Set the Content-Type HTTP header must be set to "multipart/mixed; boundary=<boundary\_identifier>".
- Set the body of the `$batch` request with individual segments that:
  - Each begin with `--batch_<boundary_identifier>`.
  - Each have the HTTP Content-Type header set to `application/http`.
  - Each have the HTTP Content-Transfer-Encoding header set to `binary`.
  - Each are then followed by the HTTP GET request in the format `GET <URI_for_the_data_endpoint> HTTP/1.1`.
  - Each can also have additional headers set for language and accepted content type.

## Request Example

```
POST api/v1/OData/$batch HTTP/1.1
Host: {jam#} [page 246].sapjam.com
DataServiceVersion: 2.0
MaxDataServiceVersion: 2.0
Content-Type: multipart/mixed; boundary=batch_12354ad7-a4rf-4b56-45tf-is47
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Members('<Member_ID>') HTTP/1.1
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Groups?$filter=substringof('r', Name)&$top=2&$format=json HTTP/1.1
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
```

```
GET Groups(<Group_ID>) HTTP/1.1
Accept-Language: zh-TW
Accept: application/json
--batch_12354ad7-a4rf-4b56-45tf-is47--
```

## \$batch Responses

If the batch request is accepted for processing:

- The server responds with an HTTP 202 Accepted message.
- The Content-Type for the response is in the form "multipart/mixed; boundary=batchresponse\_WbQa7BBJtMKBWj87c2d60g;".
- The response body will contain responses to the individual requests, separated by the boundary.

## Request with an Error

If an error occurs during the process, the batch processing stops, and the response will have sections for all the request segments up to the request that caused the error.

```
curl -i -H 'Accept: application/json'
-H 'Content-Type:multipart/mixed; boundary=batch_12354ad7-a4rf-4b56-45tf-is47'
-H 'Authorization: Bearer <!--[An OAuth2.0 access token has been removed from here.]-->'
-X POST -d "--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Members('nJlA2eTlUWlhPGI3ODnLqX') HTTP/1.1
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Groups?$filter=substringof('r', Name)&$top=2&$format=json HTTP/1.1
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Groups('NotFound_ErrorOccursAtThisRequest') HTTP/1.1
Accept-Language: zh-TW
Accept: application/json
--batch_12354ad7-a4rf-4b56-45tf-is47
Content-Type: application/http
Content-Transfer-Encoding: binary
GET Groups('AnotherValidIdButNotProcessed') HTTP/1.1
Accept-Language: zh-TW
Accept: application/json
--batch_12354ad7-a4rf-4b56-45tf-is47--" "https://{jam#}.sapjam.com/api/v1/OData/\
$batch"
```

## Response to a Request with an Error

```
HTTP/1.1 202 Accepted
```



```

X-RateLimit-Limit: 800
X-RateLimit-Remaining: 785
X-RateLimit-Reset: 2671
DataServiceVersion: 2.0
Content-Type: multipart/mixed; boundary=batchresponse_QEAXL7HqYGpUZpjBVIuaik;
charset=utf-8
Cache-Control: no-cache
X-Request-Id: effd934f0bc3a0a38d518da0e30ff570
X-Runtime: 0.463114
Date: Wed, 19 Nov 2014 18:15:28 GMT
X-Rack-Cache: invalidate, pass
Connection: close
Server: thin 1.6.1 codename Death Proof
--batchresponse_QEAXL7HqYGpUZpjBVIuaik
Content-Type: application/http
Content-Length: 3089
Content-Transfer-Encoding: binary
HTTP/1.1 200 OK
Content-Type: application/atom+xml
Content-Length: 2970
Content-Language: en
Dataverseversion: 2.0
<?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"/>
<More content has been removed to maintain readability.>
--batchresponse_QEAXL7HqYGpUZpjBVIuaik
Content-Type: application/http
Content-Length: 71636
Content-Transfer-Encoding: binary
HTTP/1.1 200 OK
Content-Type: application/atom+xml
Content-Length: 71516
Content-Language: en
Dataverseversion: 2.0
<?xml version="1.0" encoding="UTF-8"?>
<More content has been removed to maintain readability.>
--batchresponse_QEAXL7HqYGpUZpjBVIuaik
Content-Type: application/http
Content-Length: 198
Content-Transfer-Encoding: binary
HTTP/1.1 400 Bad Request
Content-Type: application/json
Content-Length: 68
Content-Language: zh-TW
Dataverseversion: 2.0
{"error":{"code":"","message":{"lang":"zh-TW","value":"參數的無效值 : Id"}}}
--batchresponse_QEAXL7HqYGpUZpjBVIuaik--

```

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know how to form [batch requests \[page 387\]](#).
- Do you understand the [request example \[page 387\]](#).
- Do you know what to expect in [batch responses \[page 388\]](#).
- Do you know how to recognize a [request with an error \[page 388\]](#).
- Do you know what to expect in a [response to a request with an error \[page 388\]](#).

If you are satisfied that you have learned what you need from this page, continue to the next page in this lesson, [Supported HTML tags \[page 390\]](#).

## 8.1.2.11 Supported HTML tags

This page lists the HTML tags that are supported in various SAP Jam Collaboration OData API inputs such as inputs for wikis, blogs, Questions, Ideas, and Discussions.

Basic HTML inputs can be either escaped or wrapped using a CDATA tag. Not all HTML is supported, for example scripts, frames and iframes, and forms are not supported, but the tags listed in the categories in the page are allowed.

### i Note

All HTML input is sanitized; only valid HTML will be accepted.

## Learning Objectives

The learning objectives for this page are:

- Know which [basic tags and structure tags \[page 390\]](#) are supported.
- Know which [formatting tags \[page 391\]](#) are supported.
- Know which [image tags \[page 391\]](#) are supported.
- Know which [table tags \[page 391\]](#) are supported.

## Basic tags and structure tags

Supported Tags	Supported Attributes
div	class height id style width
h1, h2, h3, h4, h5, h6	align clear height width
p	align clear height width style
pre	clear width wrap
ol, ul	align clear height start type width
li	align clear height type width
a	class href rel rev target title type name
hr	align clear color noshade size width
br	clear

## Formatting tags

Supported Tags	Supported Attributes
<b>span</b>	align class height id style width
<b>center</b>	align height width
<b>strong</b>	class dir id lang style title
<b>em</b>	class dir id lang style title
<b>u</b>	class dir hidden id style title
<b>s</b>	class dir hidden id style title
<b>blockquote</b>	align cite clear height type width

## Image tags

Supported Tags	Supported Attributes
<b>img</b>	align alt border class height hspace src style vspace width usemap

## Table tags

Supported Tags	Supported Attributes
<b>table</b>	align background bgcolor border bordercolor bordercolordark bordercolorlight bottompadding cellpadding cellspacing class clear cols height hspace leftpadding rightpadding rules summary toppadding vspace width style
<b>colgroup, col</b>	align bgcolor char charoff span valign width
<b>tbody</b>	align bgcolor char charoff valign
<b>th</b>	abbr align axis background bgcolor bordercolor bordercolordark bordercolorlight char charoff headers height nowrap rowspan scope valign width
<b>tr</b>	align background bgcolor bordercolor bordercolordark bordercolorlight char charoff height nowrap style valign
<b>td</b>	abbr align axis background bgcolor bordercolor bordercolordark bordercolorlight char charoff colspan headers height nowrap rowspan scope style valign width

## Learning Objectives Review

Have you accomplished the learning objectives for this page? Use the links to jump back if you want to review this material.

- Do you know which [basic tags and structure tags \[page 390\]](#) are supported?
- Do you know which [formatting tags \[page 391\]](#) are supported?
- Do you know which [image tags \[page 391\]](#) are supported?
- Do you know which [table tags \[page 391\]](#) are supported?













If you are satisfied that you have learned what you need from this page and from this tutorial, start using the SAP Jam Collaboration OData API, because this concludes the SAP Jam OData API Tutorial.

## 8.1.2.12 Supported Markdown

















This page lists the Markdown that is supported in SAP Jam Collaboration Messages.

SAP Jam follows most of the CommonMark spec, with some exceptions (for example: multiple consecutive newlines are supported instead of collapsed; emoticon support).


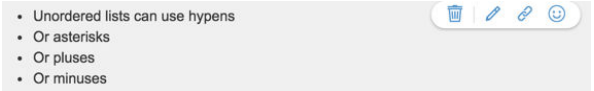
### Headers

Supported Markdown	Result
# H1	H1    
## H2	H2    
### H3	H3    

### Emphasis

Supported Markdown	Result
Emphasis, aka italics, with <code>*asterisks*</code> or <code>_underscores_</code> .	<i>italics</i>    
Strong emphasis, aka bold, with <code>**asterisks**</code> or <code>__underscores__</code> .	<b>bold</b>    
Combined emphasis with <code>**asterisks</code> and <code>_underscores_</code> .	<b><i>bold italics</i></b>    
Strikethrough uses two tildes. <code>~~strikethrough~~</code>	<del>strikethrough</del>    


## Lists

Supported Markdown	Result
<pre>1. First item in numbered list 1. Second 1. Third. Note the numbers don't actually matter</pre>	 A screenshot of a rendered numbered list with three items: "1. First item in numbered list", "2. Second", and "3. Third. Note the numbers don't actually matter". The list is displayed in a light gray box with a toolbar on the right containing icons for deleting, editing, linking, and adding a smiley face.
<pre>- Unordered lists can use hypens * Or asterisks + Or pluses - Or minuses</pre>	 A screenshot of a rendered unordered list with four items: "• Unordered lists can use hypens", "• Or asterisks", "• Or pluses", and "• Or minuses". The list is displayed in a light gray box with a toolbar on the right containing icons for deleting, editing, linking, and adding a smiley face.

## Links

Supported Markdown	Result
<pre>[Inline-style link] (http://example.com)</pre>	 A screenshot of a rendered inline-style link. The text "Inline-style link" is displayed in a light gray box with a toolbar on the right containing icons for deleting, editing, linking, and adding a smiley face.
<pre>[Reference-style link][1]  [Reference style link][hello world] aliases are case-insensitive.  [hello world]: http://sap.com [1]: http://google.com</pre>	 A screenshot of a rendered reference-style link. The text "Reference-style link" is displayed in a light gray box with a toolbar on the right containing icons for deleting, editing, linking, and adding a smiley face. Below it, the text "Reference style link aliases are case-insensitive." is displayed.

## Code and Syntax Highlighting

Supported Markdown	Result
<pre>Inline `code` has `back-ticks` around it.</pre>	 A screenshot of a rendered code block. The text "Inline code has back-ticks around it." is displayed in a light gray box with a toolbar on the right containing icons for deleting, editing, linking, and adding a smiley face. The code is highlighted with syntax highlighting: "code" is in red, "back-ticks" is in blue, and "around" is in red.

## Supported Markdown

## Result

Blocks of code are fenced with three back-ticks.

```
```javascript
var hi = "hello world";
console.log(hi);
```

no language indicated.
```
```

Blocks of code are fenced with three back-ticks.

```
var hi = "hello world";
console.log(hi);
```

no language indicated.

## Tables

## Supported Markdown

## Result

Colons can be used to align columns.

```
| Tables | Are | Cool |
| ----- | --- | ----- |
| 1 | a | $1600 |
| 2 | b | $12 |
| 3 | c | $1 |
```

There must be at least three dashes separating each header cell.

The outer pipes (|) are optional, you do not have to align the table perfectly in raw Markdown.

Colons can be used to align columns.

**Tables Are Cool**

1	a	\$1600
2	b	\$12
3	c	\$1

There must be at least three dashes separating each header cell. The outer pipes (|) are optional, you do not have to align the table perfectly in raw Markdown.

## Blockquotes

## Supported Markdown

## Result

```
> This emulates email reply text.  
> This linke continues the block quote  
above.  
  
> Another blockquote.  
  
>> But with a nested blockquote added  
with an additional '>'
```

This emulates email reply text.  
This linke continues the block quote above.  
Another blockquote.  
But with a nested blockquote added with an additional '>'

## Emoticons to Emojis

## Supported Markdown

## Result

```
Smiley :)  
Sad face :(  
Confused :/  
  
The full list of supported emoticons are here ➡
```

Smiley 😊  
Sad face 😞  
Confused 😕

## 8.1.3 Authentication and Authorization API

Client applications using the SAP Jam REST or OData APIs have three options for providing authorization and authentication for their users: an OAuth 2.0 Alias User workflow, an OAuth1.0a 3-Legged workflow, or an OAuth 2.0/SAML workflow that uses a SAML bearer assertion from a trusted identity provider to obtain an OAuth2 access token.

The OAuth 1.0a workflow is best for client applications without access to a SAML identity provider (IDP), although it requires some interaction with the end user. Both of the OAuth 2.0 workflows do not require any interaction from the user but they do require additional setup in SAP Jam by your company administrator. Additionally, there is the possibility of single-use tokens, although this approach is better suited to granting immediate, short-term access to single pages.

The following resources are available for these workflows to help get you started:

- [Alias User OAuth 2.0 Authorization Tutorial \[page 396\]](#)
- *Get an SAP Jam OAuth2 Access Token from a SAML Assertion with a CLI*
- [OAuth 1.0a Authorization Tutorial \[page 406\]](#)
- [Java Sample - jam\\_java\\_oauth1\\_client](#) ➡
- [Java Sample - jam\\_java\\_oauth1\\_hmac\\_sha1\\_client\\_sample-SHA1](#) ➡

## 8.1.3.1 Alias User OAuth 2.0 Authorization Tutorial

SAP Jam supports the use of OAuth 2.0 Authorization for Alias Users. Alias users provide an easy way to allow bots or groups of users to securely interact with SAP Jam through 3rd party applications.

### Create the Alias User OAuth 2.0 key

Perform steps 1 and 2 from [Add an OAuth Client \[page 14\]](#).

### Create an alias account

Perform steps 1 - 4 from [Alias Accounts \[page 398\]](#) and copy the alias user OAuth2 Access Token.

### Call the API with a 3rd party OData client

Add the following HTTP headers to your 3rd party OData client:

- Authorization: Bearer {Paste\_Alias\_User\_OAuth2\_Access-Token\_Here}
- Accept: application/json
- ContentType: application/json

Go to [HTTP headers \[page 244\]](#) for more information.

Make OData API calls using the following base URLs:




- **Collaboration:**
  - (Data Center) `https://{jam#}.sapjam.com/api/v1/OData/`
  - (Custom Domain) `https://{custom_domain}/api/v1/OData/`
- **Communities:**
  - (Custom Domain) `https://{custom_domain}/api/v1/OData/`

Go to [URLs and URL parameters \[page 245\]](#) for more information

### 8.1.3.1.1 Add an OAuth Client

You can authorize an external application to access the SAP Jam API by registering the application as an OAuth client.

#### To manage OAuth Clients

1. Go to the SAP Jam Admin console and select  [Integrations](#)  [OAuth Clients](#)  from the left side navigation.



OAuth Clients			
OAuth clients registered for your company.			
			<a href="#">Add OAuth Client</a>
Name	Integration URL	Date Added	
Example.com's CRM	http://sap-crm.example.com	February 27, 2015 11:46 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's ECC (SD)	https://sap-ecc-sd.example.com	April 09, 2015 02:27 PM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>
Example.com's C4C	https://sap-c4c.example.com	August 10, 2015 11:25 AM	<a href="#">View</a>   <a href="#">Edit</a>   <a href="#">Delete</a>

### OAuth Clients catalog

This page presents a catalog of previously configured OAuth Clients, with UI controls that allow you to [View](#), [Edit](#), or [Delete](#) existing OAuth Clients, or to add a new OAuth client ([Add OAuth Client](#)).

- To add an OAuth client, click [Add OAuth Client](#) at the upper right corner of the page. The [Register a new OAuth Client](#) page displays.
  - In the [Name](#) field, enter a meaningful name that allows company administrators to recognize what the client is.
  - (Optional) From the [Feed Filtering](#) dropdown menu, select either [none](#) or [SAP CRM](#).
  - In the [Integration URL](#) field, enter the URL to the client application API metadata.
  - (Optional) In the [Callback URL](#) field, enter the callback URL for the client application API calls.
  - (Optional) In the [Support URL](#) field, enter the support URL for the client application API.
  - (Optional) Select the [Can Suppress Notifications](#) checkbox to allow the suppression of notifications from external data sources that use this OAuth client. It is up to the developer of this external application integration whether they disable notifications or not, but this setting determines whether notification suppression is permitted from this external application.
  - (Optional) Select the [Can Suppress Webhooks](#) to allow the suppression of webhooks for specific OData calls in the OAuth client. It is up to the developer of this external application integration whether they disable webhooks or not, but this setting determines whether webhook suppression is permitted from this external application.
  - (Optional) In the [X509 Certificate \(Base64\)](#) text box, enter the Transport Layer Security (TLS; supersedes SSL) public key certificate string for the client application API access.
  - (Optional) The [Administrative Area](#) dropdown menu allows you to select the area in which you want this OAuth Client configuration to be available. The default is "Company", which makes it available to all groups and areas. Selecting a specific area limits the scope of the OAuth Client configuration and limits the management of that configuration to either area administrators assigned responsibility for that area or to company administrators.
  - When all of the above settings are complete, click [Save](#) to save the record and establish the trust relationship with the OAuth client application.  
You are returned to the [OAuth Clients](#) page, with the OAuth client record that you just added listed in the catalog.
- To view the information for an OAuth client, click [View](#) on the row for the OAuth Client that you want to view.  
The [OAuth Client: <OAuth\\_client\\_name>](#) page displays.  
You can either modify the information by clicking [Edit](#) or return to the [OAuth Clients](#) page by clicking [Back](#).

4. To edit an OAuth client record, either click [Edit](#) in the [OAuth Clients](#) page or in the [OAuth Client: <OAuth\\_client\\_name>](#) page.  
The [Edit your OAuth Client](#) page displays, which is effectively identical to the [Register a new OAuth Client](#) page.
  1. Make whatever changes are required.
  2. Click [Save](#) to save your changes.  
You are returned to the [OAuth Clients](#) page, with the modified OAuth client record that you just edited listed in the catalog.

## 8.1.3.1.2 Alias Accounts

Company administrators can now create up to 100 alias user (role-based or mailbox) accounts where one or more SAP Jam users can be assigned to use the alias user account in a formal capacity on behalf of their organization. An alias user account enables employees to easily identify a point of contact for specific interaction. For example, for new hire employees of a large, global organization, it is easier to contact a user named "HR Helpdesk" with HR related questions than it is to find out the name of the exact person.

Alias users can also be used as a system user (for example, program, bot) to work with other applications. In many SAP applications, this type of user is referred to as a technical user.

For more information on alias account configuration, please refer to the Developer guide topic, [Webhooks - Alias Users](#).

### Create an alias account

#### To create an alias account

1. Go to the SAP Jam Admin console and choose [Users](#) [Alias Accounts](#).
2. Click [New Alias Account](#). Complete the following steps for the Basic Profile section:
  - Below the automatically generated [Alias User ID](#), enter the [Alias Account Name](#). This name will be visible and used for interaction.
  - Enter some text to describe the purpose of the alias in the [Description](#) text box.
  - Click [Save changes](#) to save the basic profile information. You can return to this section later to enter more information after you save information in the other sections of this account configuration.
3. Complete the following steps for the Users section:
  - In the [Add users to alias account](#) text field, enter the name of each user that you want to add to the alias account. If you want to remove any of these users, then click [Remove](#) beside their name in the Users list.
  - Click [Save changes](#) to save the users information.
4. Complete the following steps for the API Access section and when the alias user is defined as a system user (e.g., bot):
  - Click [Add OAuth2 Access Token](#).
  - From the drop-down list, select the OAuth client and then click [OK](#). The token information will update and display in this section, along with the name of the OAuth client.
5. Complete the following steps for the Email Settings section:

- From the dropdown list, you can choose to enable or stop all email notifications.
  - Click [Save changes](#) to save the email settings.
6. Complete the following steps for the Org Chart section:
    - Enter the name of the Manager for the user(s) of the alias account, or select the [I have no manager](#) option.
    - Enter the names of the Direct Reports for the user(s) of the alias account, or select the [I have no direct reports](#) option.
    - Enter the names of the Assistants for the user(s) of the alias account, or select the [I have no assistant](#) option.
    - Click [Save changes](#) to save the organizational chart information.
  7. Complete the following steps for the Contact Information section:
    - Click [Add an email](#) to enter a home, work, or other email address.
    - Click [Add a phone](#) to enter a home, work, mobile, fax or other phone number.
    - Click [Add an IM](#) to enter a preferred messaging account (for example, Google Talk).
    - Enter any Mobile or LinkedIn information.
    - Click [Save changes](#) to save the contact information.

Once a user is assigned to an alias user account, they will be able to go to the gear icon above the global menu bar and select their alias user account, under the Switch User section of the menu. A banner appears above the Search field to indicate that the user is "Currently acting as an Alias User". Alias users can create groups and design group pages with their group administrator privileges.

## Edit an alias account

### To edit an alias account

1. On the Alias Accounts page, beside the alias account that you want to modify, choose [Action](#) [Edit](#).
2. Edit the basic profile, users, API access, email settings, and contact information.
3. Click [Save changes](#) to save the profile changes.

## Delete an alias account

### To delete an alias account

1. On the Alias Accounts page, beside the alias account that you want to delete, choose [Action](#) [Delete](#).
2. Click [Confirm](#) on the delete confirmation message to remove the alias account.

### 8.1.3.1.3 HTTP headers

HTTP headers are used to specify several aspects of an API call's operation.

The HTTP headers used by the SAP Jam OData API include:

- **Authorization: OAuth {OAuth1.Oa\_protocol\_parameters}** This provides user authorization using OAuth 1.0a, for example:

```
Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIHjIm",  
oauth_nonce="RPUt7ytQ9w",  
oauth_signature_method="HMAC-SHA1", oauth_timestamp="1316135320",  
oauth_version="1.0",  
oaccess_token="vbXkQoSkiqyYIxiI2u", oauth_signature="P5scFJf6CZlBBMELB9kb  
%2FvM0ktQ%3D"
```

- **Authorization: Bearer {OAuth\_bearer\_token}** This provides user authorization using OAuth 2.0, for example:

```
Authorization: Bearer As3UvIaYEvdXoeREtmSz3qeCpnNvrrHZhVMswcBV
```

- **Accept: {application/json|application/atom+xml}** This header allows you to specify what is an acceptable response to your API call: the XML or JSON format. This option is preferred to using a query option of `?$format=json`. Note that the default format for OData API calls is XML, so that does not need to be specified unless you want to use JSON. To do so, specify that responses be in JSON format in the HTTP Accept header. For example:

```
Accept: application/json
```

The XML option, while not required, would be:

```
Accept: application/atom+xml
```

- **ContentType: {MIME\_type}** Informs the API that you are sending content in a specific format in the request. This must be specified as the MIME type for the content that you are submitting. Valid options include:
  - `application/json`
  - `application/atom+xml`
  - `text/html`
  - `text/html;type=wiki`
  - `text/html;type=blog`
  - `image/png`
  - `image/jpg`
  - `video/mp4`
  - `application/vnd.ms-powerpoint`
  - `application/vnd.openxml-formats-officedocument.presentationml.presentation`

For example:

```
ContentType: image/png
```

The above examples are all valid, but the list of supported MIME types is far from complete.

- **SAP-JAM-CONTENT-ADMINISTRATION: {true}** This header allows company administrators to enable content administration on their company for this request. With content administration enabled, company

administrators can perform CRUD operations to all groups and content within their company regardless of group membership.

- **Example 1** - Show all Content Items in a group that the company administrator is not a member of.

```
Request Url: GET {{api_url}}/Groups('{{Id}}')/AllContentItems
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

- **Example 2** - Search for all Content Items in a group that the company administrator is not a member of.

```
Request Url: GET {{api_url}}/Search?Query='{{search_item}}'
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

- **Example 3** - Delete an item in a group that the company administrator is not a member of.

```
Request Url: DELETE {{api_url}}/
ContentItems(Id='{{Id}}',ContentItemType='{{Type}}')
Header: Authorization: Bearer {OAuth_bearer_token}
Header: SAP-JAM-CONTENT-ADMINISTRATION: true
```

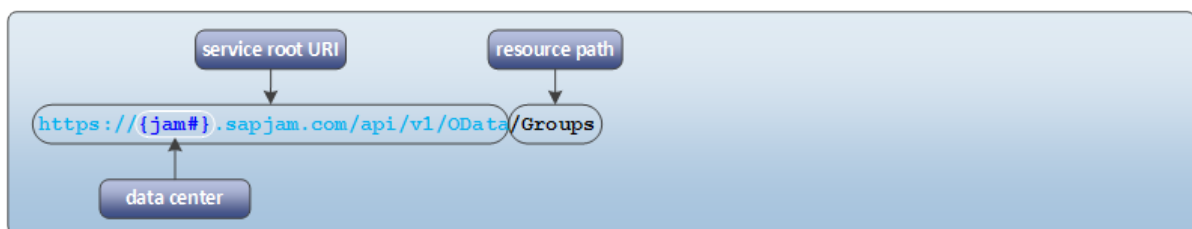
- **Slug: {file\_name\_without\_extension}** "Slug" is an HTTP header used in the SAP Jam Collaboration OData API as the name of the file used in SAP Jam. In an upload (a POST), the Slug header would be set as "profilePhoto" if that is the name you want the resource to appear as in SAP Jam. In an update (a PATCH) or a download (a GET), the Slug header would be set as "profilePhoto" if you want to update (re-upload) or download a file of that name in SAP Jam Collaboration. For example:

```
Slug: profilePhoto
```

### 8.1.3.1.4 URLs and URL parameters

OData APIs are accessed via a URL that calls a specific endpoint and has the required URL parameters properly set. There are different forms of URL for each of several specific types of operation.

The most basic API call specifies an entity, but not a specific resource. This is done for POST operations where a new resource has not yet been assigned a unique ID by SAP Jam, and in collection-valued GET requests, from which you want to retrieve a collection (or feed) of the available resources of that entity type. The basic parts of such a simple request (get me the available groups) are:



## service root

Identifies the data center or server of an OData service and location of the API in that data center or server. A generalized form of the service root URI is shown for the SAP Jam OData API.

## resource path

Identifies the resource to be interacted with. In the preceding diagram, the resources involved are one or more ContentItems in the SAP Jam Group that is specified by its unique ID.

Note that the combination of the HTTP method and a generalized form of the resource path is used as the signature of the endpoints in the SAP Jam OData API.

## {jam#} usage in the URLs (data centers)

SAP Jam Collaboration services are located in data centers in various locations around the world. Your organization's SAP Jam service will be at one of these locations, which must appear in the URL of your API calls, and which is indicated in this documentation by {jam#}, which is always in the form "jam#", in which "#" represents the 1 or 2 digit number of the data center in which your organization's SAP Jam instance is hosted. View the URL of any of your organization's SAP Jam pages to find your SAP Jam data center number.

## Unique IDs

To identify a specific resource, the SAP Jam OData API uses 22-character unique identifiers for the various accessible objects in SAP Jam Collaboration. This is shown in the following URL diagram:



For most entities, the unique ID alone is sufficient to identify a specific resource, but for some entities two attribute-value pairs are required. These one or two properties are commonly termed the "key" for the resource, and the key value, or values, must be set in the URL of an API call to indicate exactly which resource to perform the API call operation on. The entities that require two attribute-value pairs to identify a specific resource, and the attributes that they use are as follows:

- `ContentItem(Id='{Id}',ContentItemType='{ContentItemType}')`, where the '{ContentItemType}' can be "Page", "BlogEntry", "Document", "Tool", or "Poll".
- `ContentListItem(Id='{Id}',ContentListItemType='{ContentListItemType}')`, where the '{ContentListItemType}' can be "Page", "BlogEntry", "Document", "Tool", or "Poll".

- `EventResponse (RepondentId=' {RespondentId} ',EventId=' {EventId} ')`
- `Folder (Id=' {Id} ',FolderType=' {FolderType} ')`, where the `{FolderType}` can be "Folder" or "PrivateFolder".
- `GroupExternalObject (GroupId=' {GroupId} ',ExternalObjectId=' {ExternalObjectId} ')`
- `GroupMembership (GroupId=' {GroupId} ',MemberId=' {MemberId} ')`
- `GroupTemplate (Id=' {Id} ',GroupTemplateType=' {GroupTemplateType} ')`, where the `{GroupTemplateType}` can be "system" or "custom".
- `Kudo (Id=' {Id} ',KudoType=' {KudoType} ')`, where the `{KudoType}` can be "system" or "custom".
- `ObjectReference (Id=' {Id} ',Type=' {Type} ')`, where the `{Type}` can be "Member", "Group", "WallComment", "Event", "Task", "MemberKudo", "Comment", "FeedEntry", "ForumItem", or "ContentItem".
- `TaskAssignment (AssigneeId=' {AssigneeId} ',TaskId=' {TaskId} ')`
- `ThumbnailImage (Id=' {Id} ',ThumbnailImageType=' {ThumbnailImageType} ')`, where currently the only valid value for `{ThumbnailImageType}` is "48x48".
- `ThumbnailKudoImage (Id=' {Id} ',ThumbnailImageType=' {ThumbnailImageType} ')`, where currently the only valid value for `{ThumbnailImageType}` is "48x48".

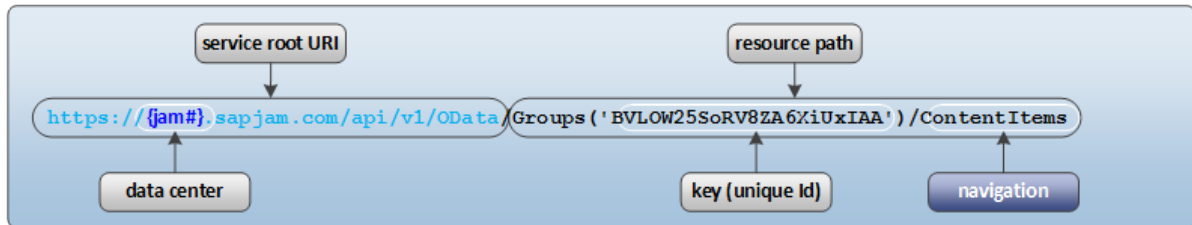
For the specification reference, see <http://www.odata.org/documentation/odata-version-3-0/common-schema-definition-language-csdl/#csdl6.2>.

## navigations

Navigations are URL extensions from the base entity to closely related entities. They can be named according to their "role", for example, one navigation from the Group entity is to a "Creator", which is the role, but which returns the "Member" information on the individual who created the group. Every entity has a predefined set of navigations. For example, the Group entity has the following navigations:

- Role: AllContentItems (Entity: ContentItem)
- Role: AllDiscussions (Entity: Discussion)
- Role: AllFolders (Entity: Folder)
- Role: AllIdeas (Entity: Idea)
- Role: AllQuestions (Entity: Question)
- Role: ContentItems (Entity: ContentItem)
- Role: ContentListItems (Entity: ContentListItem)
- Role: Creator (Entity: Member)
- Role: FeaturedExternalObjects (Entity: ExternalObject)
- Role: FeedEntries (Entity: FeedEntry)
- Role: Folders (Entity: Folder)
- Role: Forums (Entity: Forum)
- Role: Memberships (Entity: GroupMembership)
- Role: ParentGroup (Entity: Group)
- Role: PrimaryExternalObject (Entity: ExternalObject)
- Role: PrimaryGadgetObject (Entity: GroupGadgetObject)
- Role: ProfilePhoto (Entity: Image)
- Role: SubGroups (Entity: Group)

- Role: Tasks (Entity: Task)
- Role: Template (Entity: GroupTemplate)
- Role: UpcomingEvents (Entity: Event)



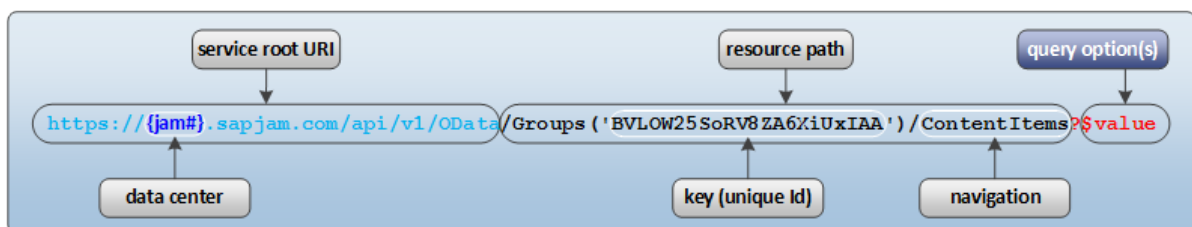
## query options

Query options are URL parameters that are added to the end of the URL.

There are three general types of query options:

- **System query options**: are indicated with a preceding dollar sign "\$". In the diagram above, the \$value segment is a system query option.
- **Custom query options**: are not used in the SAP Jam OData API, and so they are not discussed any further here.
- **Service operation parameters**: are the required parameters for service operations, which are discussed at greater length below.

The SAP Jam Collaboration OData API supports OData system query options with a number of limitations to ensure system responsiveness.



- There can be at most one Collection-valued navigation in an SAP Jam OData request.
- There can be at most 3 \$expand operations in one API call. Using \$expand on a Collection valued navigation of a Collection is not allowed.  
See in "OData Query Parameters", Use \$expand to include details on any property for which there are available navigations.
- Using \$filter is currently only supported for select properties of the following entities:
  - **ContentItem**: ContentItemType
  - **ContentListItem**: Name
  - **ExternalObject**: Exid, ObjectType
  - **Folder**: Name
  - **FeedEntry**: Read
  - **ForumItem**: ForumItemType
  - **Group**: Id, Name, IsActive, GroupType



- **GroupExternalObject:** LinkType
- **GroupMembership:** MemberType
- **Idea:** Status
- **Notification:** Category, EventType
- **Question:** HasBestAnswer
- **Task:** IsOverdue
- **TaskAssignment:** Status

See in "OData Query Parameters", Use `$filter` to limit the entries returned according to their content.

- SAP Jam will never return arbitrarily large collections. Either server-driven paging (use `$skiptoken`) or client-driven paging (use `$skip`) will apply. Certain endpoints only support server-driven paging (`$skiptoken`). In general, a maximum of 20 items will be returned, unless a value of `$top` is specified (and supported), in which case the maximum value is 100.

See in "OData Query Parameters":

- Use `$skip` to offset the set of entries returned.
- Use `$skiptoken` to offset the set of entries returned.
- Use `$top` to set the number of entries returned.

- For FeedEntries and Notifications API calls, `$skip`, `$top`, and `$count` are not available. Also, `$count` is not available for GroupMembership API calls, and `$orderby` is not available for FeedEntries and Notifications, but it will be selectively available where the items are sortable in the SAP Jam user interface.

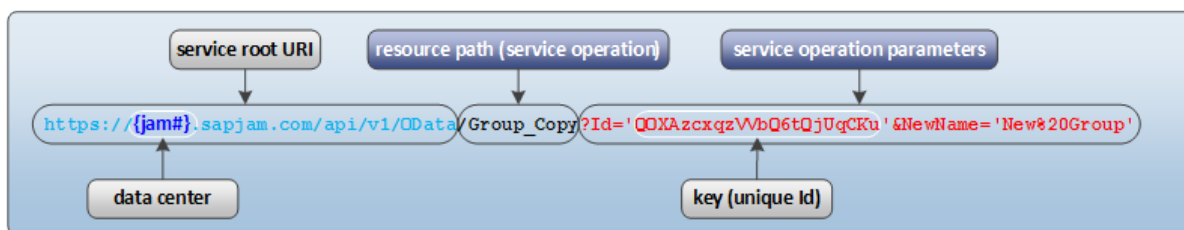
See in "OData Query Parameters":

- Use `$skip` to offset the set of entries returned.
- Use `$top` to set the number of entries returned.
- Use `$count` to discover how many entries there are.
- Use `$orderby` to set the criteria that the returned entries are ordered by.

See the OData v.2 Specification's discussion of [OData System Query Options](#) .

## Service Operations

In practice, a service operation is any endpoint that performs a required task that does not fit with the usual POST, GET, PATCH, or DELETE operations. In SAP Jam Collaboration's implementation, service operations are signified by using underscores between words in the endpoint name rather than showing the name in camel-case. There are, however, at least four exceptions: `[GET] /Self`, `[GET] /Company`, `[GET] /Search`, and `[GET] /SearchSummary`.

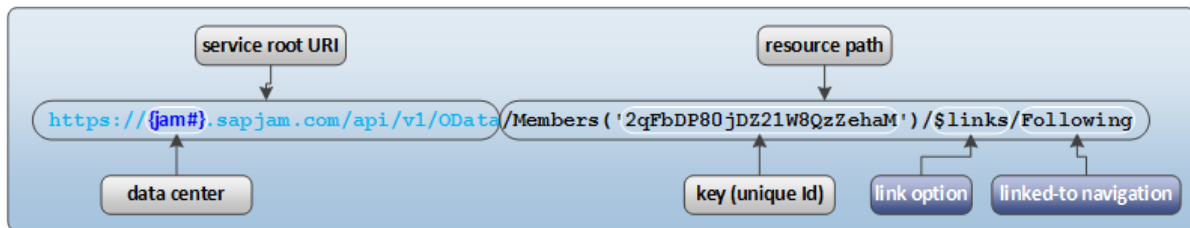


Note that if you don't find sufficient information about a service operation in this *SAP Jam Collaboration API Tutorial* or the *SAP Jam Collaboration API Reference*, then you can check the `$metadata` file. Each service operation is described to some extent in that file.

For the technical specification for a service operation, see section 2.13 of the OData 2.0's [Operations](#) page.

## Links between entries

The OData specification allows for the creation of links between entries.



For example, in the SAP Jam Collaboration OData API, currently logged-in Member can be set as "Following" another specified Member by setting a "\$link" from the Member to be followed to "Following", which will take the Id of the currently logged-in Member as the "follower". For example:

```
https://{jam#}.sapjam.com/api/v1/OData/Members('2qFbDP80jDZ21W8QzZehaM')/$links/Following
```

There are a small number of such operations available in the SAP Jam OData API, but they are important operations. They include:

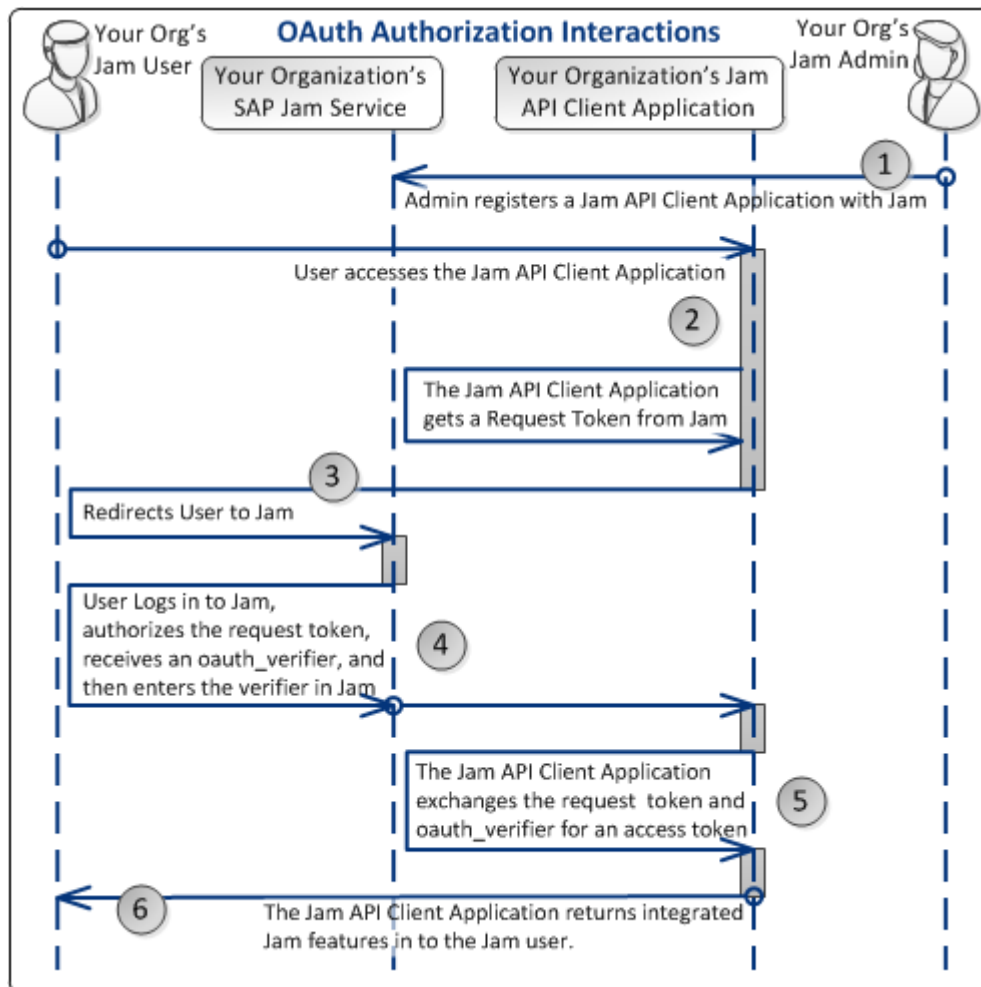
- [POST] /Members('{id}')/\$links/Following
- [DELETE] /Members('{id}')/\$links/Following('{id1}')
- [POST] /Questions('{id}')/\$links/BestAnswer
- [DELETE] /Questions('{id}')/\$links/BestAnswer
- [POST] /Groups('{id}')/\$links/FeaturedExternalObjects
- [DELETE] /Groups('{id}')/\$links/FeaturedExternalObjects('{id1}')
- [POST] /Events('{id}')/\$links/Invitees
- [POST] /Tasks('{id}')/\$links/Attachments
- [DELETE] /Tasks('{id}')/\$links/Attachments('{id1}')
- [POST] /Tasks('{id}')/\$links/PendingFollowers

See sections [2.9 Manipulating Links](#), [2.10 Creating Links between Entries](#), [2.11 Removing Links between Entries](#), and [2.12 Replacing Links between Entries](#) in the Operations page of the OData 2.0 specification.

## 8.1.3.2 OAuth 1.0a Authorization Tutorial

SAP Jam supports 3-legged OAuth 1.0a authorization, which provides a mechanism for SAP Jam-integrated applications to get explicit authorization from an end user for that application to present SAP Jam content. The steps required to use this workflow are documented in this section.

The workflow for this authorization method is as follows:



1. Your organization's SAP Jam administrator registers your organization's SAP Jam API client application with SAP Jam.
2. When the user views the SAP Jam API client application, it calls SAP Jam to obtain a request token (POST /oauth/request\_token).
3. The SAP Jam API client application then redirects the end user to SAP Jam.
4. The end user logs into SAP Jam and is presented with an authorization request to allow the SAP Jam API client application to make calls to SAP Jam on their behalf. If they accept, they receive an oauth\_verifier that they enter into SAP Jam.
5. The SAP Jam API client application then exchanges the request token and the oauth\_verifier for an authorized access token (POST /oauth/access\_token).
6. Subsequently, the SAP Jam API client application can make public REST API calls on behalf of the end user, allowing the user to see SAP Jam elements in the SAP Jam API client application.

The SAP Jam OAuth API also provides a utility call that allows you to validate an authorized access token.

It is also important to be able to revoke an authorized access token.

## 8.1.3.2.1 1. Register your Auth Client

A client application must be registered in SAP Jam before it can use SAP Jam as an OAuth provider.

To register a new OAuth client, do the following:

1. Log on to your company's SAP Jam Collaboration account as an administrator.
2. At the top of the page, click the account name then click [Admin](#).
3. On the left side of the page, click [OAuth Clients](#).
4. At the bottom of the page, click [Add OAuth Client](#).

There are various fields, but only the following are relevant to the OAuth workflow:

- Company Name: Example Corporation
- Integration URL: <http://example.com>
- Callback URL: <https://apis.example.com>

### i Note

- It is recommended that the Callback URL be https.
- If the client app does not support callbacks, as may be the case with a mobile application or a desktop application, then the Callback URL should be left blank. This affects what happens when the end user authorizes the client app on the SAP Jam Collaboration website.

5. Click [Save](#).

SAP Jam generates a client application key and client application secret unique to the registered company. It is important that the client application key is kept secure and private, because it is essentially like the client application's password.

The sample client app created in step 4 above has the following data, which is used throughout this document:

```
OAuth details for Example OAuth Client Application
Consumer Key: J2iLG8a8xZPtqBifHjIm
Consumer Secret: puTHhU5y8upl9y8fJr25rTAlLXwZEKUKqib89bwJ
Request Token URL: https://samplejamserver.com/oauth/request_token
Access Token URL: https://samplejamserver.com/oauth/access_token
Authorize URL: https://samplejamserver.com/oauth/authorize
We support the PLAINTEXT, HMAC-SHA1 and RSA-SHA1 signature methods in SSL mode
```

### 8.1.3.2.1.1 1.1 Signature Methods

This page discusses: supported signature methods, security considerations, details on the use of the available signature methods, and the advantages and disadvantages of each.

## Supported signature methods

The PLAINTEXT, HMAC-SHA1, and RSA-SHA1 signature types are supported.

## Performance considerations

In the vast majority of cases, performance should not be a factor when deciding which OAuth signature type to use. Performance details are as follows:

- RSA-SHA1 is slower than HMAC-SHA1 which is slower than PLAINTEXT.
- Most of the delay of RSA-SHA1 is on the client side, which is due to the overhead of signing the request.
- The overhead of both the client side and provider side of processing the OAuth request is under 9 ms per request. For most REST API requests, which are on the order of 500+ ms, this represents a small fraction of the time for the request.

## Security considerations

From a security point of view, SAP Jam Collaboration's recommendations are as follows:

- Use RSA-SHA1 for OAuth client applications that are able to keep user-level as well as application-level secrets, such as a secured web application (secure server to secure server calls).
- Use HMAC-SHA1 for OAuth client apps that are able to keep user-level secrets, but not application-level secrets, such as a mobile client or a desktop client. However, in some cases, RSA-SHA1 may still be better: see the next section for details.
- Do not use OAuth authentication when the client app cannot keep any sort of secret.
- Use HMAC-SHA1 instead of PLAINTEXT if possible. Although PLAINTEXT is secure when appropriately used under SSL, HMAC-SHA1 is more secure and has equivalent usage characteristics in other respects.

## Details and discussion

OAuth client app credentials consist of a client application key and a client application secret, which in the OAuth protocol are known as the `oauth_consumer_key` and `oauth_consumer_secret`, respectively. The client application secret should be kept secret by the client app. There is one secret per OAuth application, regardless of the number of client app users. The client application key and secret are randomly generated UUIDs, generated using a cryptographically strong pseudo-random number generator. The key is represented by 20 Base64 characters (120 bits) and the secret by 40 Base64 characters (240 bits). A sample client application key and secret are shown below:

```
oauth_consumer_key: fb8yyZdZ58WipQLy1ZYX
oauth_consumer_secret: 50DFsABZBNUJl24hdEztjk2l1ddLX7up4PIYDvxxk4
```

OAuth token credentials consist of a token key and a token secret, which in the OAuth protocol are known as the `oauth_token` and `oauth_token_secret`, respectively. There are one or more tokens per OAuth application, as there is one token per client app user. The token key and secret are randomly generated UUIDs, generated using a cryptographically strong pseudo-random number generator. The key is represented by 20 Base64 characters (120 bits) and the secret by 40 Base64 characters (240 bits). These tokens can be stored by the client app to avoid having to request end user authorization every time the user wants to access SAP Jam Collaboration resources.

The RSA-SHA1 signature method does not use either the client application key secret or the OAuth token secret described above. A client app is configured to use the RSA-SHA1 signature method by configuring the X-509 certificate when registering the client app at the chosen OAuth provider.

#### **i Note**

Under SAP Jam Collaboration's implementation, an OAuth provider configured to use RSA-SHA1 will only accept RSA-SHA1 signed requests.

For an RSA-SHA1 OAuth client, there is no need to store the client application secret or the OAuth token secret. The RSA private key used to sign requests by the client must be stored securely as an application-level secret. In particular, it should not be accessible to users of the client application.

When an RSA-SHA1 OAuth client makes regular 3-legged OAuth calls, it makes use of the RSA private key, the `oauth_consumer_key`, and the `oauth_token` to sign the request. However, if the OAuth client cannot be relied on to keep application-level secrets (such as the RSA private key), then the HMAC-SHA1 authentication method is recommended. This is because RSA-SHA1 uses the 120-bit `oauth_token` to secure the request, while HMAC-SHA1 uses the 240-bit `oauth_token_secret`.

## **Advantages and disadvantages**


The main advantage of the RSA-SHA1 signature method is that it is not based on shared secrets. The client does not need to worry about their `oauth_consumer_secret` or `oauth_token_secret` being compromised at the provider site. For example, it is impossible for an employee in the operations team at the provider site to generate OAuth requests for the client.

Another advantage of RSA-SHA1 is that the client can easily change its configuration to update to a new RSA private key by uploading a new X-509 certificate to the provider. For the other signature methods, it is currently not possible to change the `oauth_consumer_secret` without re-registering the client application.

#### **i Note**

This is not a fundamental restriction of OAuth; the feature to generate a new secret has simply not been implemented. Currently, the client app would need to be deleted and re-registered at the OAuth client provider, and users would need to reauthorize their access tokens the next time they use the application.

An advantage of the RSA-SHA1 and HMAC-SHA1 signature methods is that all query parameters and any parameters in the Authorization header are included in computing the signature. This means that if any of these parameters are tampered with, OAuth validation will fail. Also, in addition to authorizing OAuth 1.0a access tokens, users can view the applications they have authorized and unauthorize them.

A disadvantage of the PLAINTEXT signature method is that the `oauth_consumer_secret` and `oauth_token_secret` values are visible and unencrypted within the `oauth_signature` parameter. This makes it easier for these values to be compromised in the case of poor logging practices - see <http://tools.ietf.org/html/rfc5849#section-3.4.4>  for more information. While in theory, proper adherence to SAP security standards on logging means this will not be a problem, it is better to have redundant layers of security, particularly when the solution to this problem (using a different signature method) is simple.

## i Note

Regardless of the chosen signature method, client apps should store the authorized access token. This means users will not be repeatedly asked to authorize the client app.

### 8.1.3.2.2 2. Get a Request Token from Jam

After the client application has been registered as an OAuth provider and that a client application key (oauth\_consumer\_key) and client application secret (oauth\_consumer\_secret) have been generated, you can get a request token using [POST] /oauth/request\_token.

The following procedure requires that the client application has been registered as an OAuth provider and a client application key (oauth\_consumer\_key) and client application secret (oauth\_consumer\_secret) have been generated.

1. The client app calls SAP Jam Collaboration to obtain a request token:

POST /oauth/request\_token/

Using the sample client application key J2iLG8a8xZPtqBIfHjIm and client application secret puTHhU5y8upl9y8fJr25rTAlLXwZEKUkQib89bwJ, a trace of the HTTP request and response is as follows:

```
> POST /oauth/request_token HTTP/1.1
> User-Agent: curl/7.20.1 (i686-pc-cygwin) libcurl/7.20.1 OpenSSL/0.9.8r zlib/
1.2.5 libidn/1.18 libssh2/1.2.5
> Host: sapjam.com
> Accept: */*
> Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIfHjIm",
oauth_nonce="y-s52YCzMQ",
    oauth_signature_method="HMAC-SHA1", oauth_timestamp="1316126857",
oauth_version="1.0",
    oauth_callback="oob", oauth_signature="mdGROILktVrGG6bvB%2FJd4hK0jSo%3D"
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Cache-Control: no-store, private
< Expires: -1
< Vary: *
< X-Runtime : 27
< ETag: "a65faf922b1c155b71814eff0b3c1152"
< Set-Cookie: _12sprints_srv=m12; path=/; secure; HttpOnly
< Set-Cookie: _cstar_session=914aedef444e795376a223735aa42c7ab; path=/;
secure; HttpOnly
< Content-Type: text/html; charset=utf-8
< Content-Length: 122
< Date: Thu, 15 Sep 2011 22:47:48 GMT
< Connection: close
< Server: SAP LJS 1.0.0
< Set-Cookie: BIGipServerstage.sapjam.com-443-pool=200941484.20480.0000;
path=/
<=====[ Response Below ]=====
oauth_token=39b1abH5FJZgXwZ2lnoq&oauth_token_secret=A6vIuS15Bps2qbH7GFD1d1UQMF
SgNP9J7GlgilEn&
    oauth_callback_confirmed=true
```

Details are as follows:

- The request must be an HTTP POST with content type application/x-www-form-urlencoded

- All of the following OAuth parameters are required:

Parameter	Description
oauth_consumer_key	The client app key, obtained when the client app was registered with the OAuth provider.
oauth_nonce	A single-use randomly generated UUID string.
oauth_signature_method	The chosen signature method (PLAINTEXT, HMAC-SHA1, or RSA-SHA1). We recommend using either the HMAC-SHA1 or RSA-SHA1 signature methods, but not PLAINTEXT.
oauth_timestamp	The current UNIX timestamp.
oauth_version	The version of OAuth being used. This must be 1.0, although this is actually 1.0a, an important security update to 1.0.
oauth_callback	The Callback URL. <ul style="list-style-type: none"> <li>○ If this parameter value is oob, then the Callback URL that was provided when the client app was registered is used.</li> <li>○ If a Callback URL was provided when the client app was registered, but this parameter value is not oob, then the URL in the parameter must be within the same top-level domain as the provided URL. For example, <code>https://apis.example.com/callback</code> is valid, but <code>https://apis.somewhereelse.com</code> is not. The response code, 401 Unauthorized, is displayed if an invalid URL is used.</li> </ul>
oauth_signature	A signature generated in accordance with the specified oauth_signature_method.

- The above parameters can be supplied in the Authorization header (as in the above example) or as query parameters in the POST request body, as specified in section 5.2 of the OAuth spec.
- If successful, the request returns HTTP 200 OK and a request token key and secret. In the above example, the key is `39b1abH5FJZgxwZ2lnoq` and the secret is `A6vIuS15Bps2qbH7GFD1d1UQMFSgNP9J7GlgilEn`.

#### i Note



The parameter names `oauth_token` and `oauth_token_secret` can be overloaded to mean either a request token key and secret or an access token key and secret, depending on the context.

- The request token secret should be kept secret by the client app. It should also be stored by the client app to avoid having to request end user authorization every time the user wants to access SAP Jam Collaboration resources.

### 8.1.3.2.3 3. Redirect the User to SAP Jam

Once the client application has acquired a request token, redirect the end user to SAP Jam Collaboration.

To redirect the end user to SAP Jam Collaboration, do the following:

1. Using the sample request token key `39b1abH5FJZgxwZ2lnoq` from Step 2, redirect the user to: [https://demojam.com/oauth/authorize?oauth\\_token=39b1abH5FJZgxwZ2lnoq](https://demojam.com/oauth/authorize?oauth_token=39b1abH5FJZgxwZ2lnoq)  

The user is then prompted to authorize the client app to access their SAP Jam Collaboration account and make REST API calls on their behalf.



## 8.1.3.2.4 4. User authorizes the client application

Once the end user has been redirected to SAP Jam Collaboration, the user must authorize their access to the client application.

When the user is redirected to SAP Jam Collaboration, they must log in using their SAP Jam credentials, and then they are shown the following prompt:

```
Authorize access to your account
Would you like to authorize Example OAuth Client Application (http://
example.com) to access your account?
```

The Company Name and Integration URL that are displayed are the values entered when the client application was registered with the OAuth provider.

1. The user can choose to:
  - Deny the request.  
Example Corporation will not be notified.
  - Authorize the request.  
SAP Jam will redirect the user to:

```
https://apis.example.com/?
oauth_token=39b1abH5FJZgxwZ2lnoq&oauth_verifier=HN7f2t1Z9ndgLqzqs8uu
```

Element	Description
https://apis.example.com/	The Callback URL entered when the client app was registered with the OAuth provider.
oauth_token	The request token key from Step 2.
oauth_verifier	Used in the following steps. This value ensures that the user who authorized the client app on SAP Jam Collaboration is the same person as the user of the client app requesting access to the SAP Jam Collaboration user's protected resources.

**i Note**  
This is the key difference between OAuth versions 1.0 and 1.0a.

If the client app's Callback URL was left blank and the `oauth_callback` parameter was set to `oob` when the client app called SAP Jam to get a request token, then SAP Jam Collaboration will not callback to the client application if the user authorizes access. Instead, SAP Jam Collaboration will display a message that the user authorized the access, along with a verification code for the user to enter when they return to the client app.

## 8.1.3.2.5 5. Request Token to Access Token

When the user authorizes access to the client application in SAP Jam Collaboration, the client application must exchange the request token for an authorized access token.

To have the client app exchange the request token obtained in Step 2 for an authorized access token, the following steps must be performed:

1. Use the verifier obtained in Step 4, as well as the client application key and secret.

```
POST /oauth/access_token
```

Using the sample client application key J2iLG8a8xZPtqBIfHjIm, client application secret puTHhU5y8upl9y8fJr25rTAlLXwZEKUKQib89bwJ, request token key 39b1abH5FJZgxwZ2lnoq, request token secret isA6vIuS15Bps2qbH7GFD1d1UQMFSgNP9J7GlgilEn, and verifier HN7f2t1Z9ndgLqzqs8uu, a trace of the HTTP request and response is:

```
> POST /oauth/access_token HTTP/1.1
> User-Agent: curl/7.20.1 (i686-pc-cygwin) libcurl/7.20.1 OpenSSL/0.9.8r zlib/
1.2.5 libidn/1.18 libssh2/1.2.5
> Host: jamdemo.com
> Accept: */*
> Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIfHjIm",
oauth_nonce="C9Rf4I-piw",
    oauth_signature_method="HMAC-SHA1" , oauth_timestamp="1316134422" ,
oauth_version="1.0",
    oauth_token="39b1abH5FJZgxwZ2lnoq", oauth_verifier="HN7f2t1Z9ndgLqzqs8uu",
    oauth_signature="yUYSVXsjIIgr9hZ3YsSQBM453lY%3D"
> Content-Type: application/x-www-form-urlencoded
>
< HTTP/1.1 200 OK
< Cache-Control: no-store, private
< Expires: -1
< Vary: *
< X-Runtime : 44
< ETag: "9d5f14e5df3e173a08cc5b84140cd0ee"
< Set-Cookie: _12sprints_srv=m13; path=/; secure; HttpOnly
< Set-Cookie: _cstar_session=55db5ac7072b9e48185e834c9234d33b; path=/;
secure; HttpOnly
< Content-Type: text/html; charset=utf-8
< Content-Length: 92
< Date: Fri, 16 Sep 2011 00:53:52 GMT
< Connection: close
< Server: SAP LJS 1.0.0
< Set-Cookie: BIGipServerstage.sapjam.com-443-pool=217718700.20480.0000;
path=/
<=====[ Response Below ]=====
oauth_token=vbXkQoSkiqYIXwI2I2u&oauth_token_secret=pYOz16l9UjgZddj9twsoD8ptWK
Q4LxrQi24xE2sA
```

Details are as follows:

- The request must be an HTTP POST with content type application/x-www-form-urlencoded.
- All of the following OAuth parameters are required:

Parameter	Description
oauth_consumer_key	The client app key, obtained when the client app was registered with the OAuth provider.
oauth_nonce	A single-use randomly generated UUID string.
oauth_signature_method	The chosen signature method (PLAINTEXT, HMAC-SHA1, or RSA-SHA1).
oauth_timestamp	The current UNIX timestamp.
oauth_version	The version of OAuth being used.

Parameter	Description
oauth_token	The request token key from Step 2.
oauth_verifier	The verifier from Step 4.
oauth_signature	The client app secret, obtained when the client app was registered with the OAuth provider.

- The above parameters can be supplied in the Authorization header (as in the above example) or as query parameters in the POST request body, as specified in section 5.2 of the OAuth spec.
- If successful, the request returns HTTP 200 OK and an authorized access token key and secret. In the above example, the key is `vbXkQoSkiQyYI2I2u` and the secret is `pYOz16l9UjgZddj9twsOD8ptWKQ4LxrQi24xE2sA`.

### Note

The parameter names `oauth_token` and `oauth_token_secret` can be overloaded to mean either a request token key and secret or an access token key and secret, depending on the context.

- The access token secret should be kept secret by the client app. It should also be stored by the client app to avoid having to request end user authorization every time the user wants to access SAP Jam Collaboration resources.

## 8.1.3.2.6 6. Make Public Jam API Calls

This page in the description of how to use OAuth 1.0a authorization demonstrates the use of an API call in SAP Jam Collaboration.

For this example, we will use `/v1/activities?page=1&page_size=5` to get the user's activities. However, any public API call in SAP Jam Collaboration can use OAuth authentication.

1. Use the sample client application key `J2iLG8a8xZPtqBIfHjIm`, client application secret `ispuTHhU5y8upl9y8fJr25rTAlLXwZEKUKib89bwJ`, access token `vbXkQoSkiQyYI2I2u`, and access token secret `pYOz16l9UjgZddj9twsOD8ptWKQ4LxrQi24xE2sA`, a trace of the HTTP request and response is as follows:

```
> GET /v1/activities?page=1&page_size=5 HTTP/1.1
> User-Agent: curl/7.20.1 (i686-pc-cygwin) libcurl/7.20.1 OpenSSL/0.9.8r zlib/
1.2.5 libidn/1.18 libssh2/1.2.5
> Host: demojam.com
> Accept: */*
> Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIfHjIm",
oauth_nonce="RPUt7ytQ9w",
    oauth_signature_method="HMAC-SHA1", oauth_timestamp="1316135320" ,
oauth_version="1.0",
    oauth_token="vbXkQoSkiQyYI2I2u", oauth_signature="P5scFJf6CZlBBMELB9kb
%2FvM0ktQ%3D"
> Content-Type: text/plain
>
< HTTP/1.1 200 OK
< Cache-Control: no-store, private
< Expires: -1
< Vary: *
< X-Runtime : 1203
```

```
< ETag: "258cfbdc42df3b2076aa87a1912f58ac"
< Set-Cookie: _12sprints_srv=m12; path=/; secure; HttpOnly
< Set-Cookie: _cstar_session=31e8fdfac7836cfbf9aed88bcc9eaa17; path=/;
secure; HttpOnly
< Content-Type: application/xml; charset=utf-8
< Content-Length: 16908
< Date: Fri, 16 Sep 2011 01:08:50 GMT
< Connection: close
< Server: SAP LJS 1.0.0
< Set-Cookie: BIGipServerstage.demojam.com-443-pool=217718700.20480.0000;
path=/
<=====[ Response Below ]=====
<?xml version="1.0" encoding="UTF-8" ?>
<activities total_number_of_activities="439">
<activity ... (omitted)
```

Details are as follows:

- All of the following OAuth parameters are required:

Parameter	Description
oauth_consumer_key	The client app key, obtained when the client app was registered with the OAuth provider.
oauth_nonce	A single-use randomly generated UUID string.
oauth_signature_method	The chosen signature method (PLAINTEXT, HMAC-SHA1, or RSA-SHA1).
oauth_timestamp	The current UNIX timestamp.
oauth_version	The version of OAuth being used.
oauth_token	The authorized access token from Step 5.
oauth_signature	The client app secret, obtained when the client app was registered with the OAuth provider.

- The above parameters can be supplied in the Authorization header (as in the above example) or as query parameters in the POST request body, as specified in section 5.2 of the OAuth spec.

### 8.1.3.2.7 Validate an Authorized Access Token

You can test that the API call checks if the access token is valid and working correctly by performing a [GET] / oauth/test\_request.

This is a test API call that checks if the access token is valid and working correctly without making unauthorized calls on behalf of the user.

1. Use GET /oauth/test\_request.

Using the sample client application key J2iLG8a8xZPtqBIfHjIm, client application secret puTHhU5y8upl9y8fJr25rTALXwZEKUKqib89bwJ, access token key vbXkQoSkiQyYI2I2u, and access token secret pY0z16l9UjgZddj9tws0D8ptWKQ4LxrQi24xE2sA, a trace of the HTTP request and response is as follows:

```
> GET /v1/activities?page=1&page_size=5 HTTP/1.1
> User-Agent: curl/7.20.1 (i686-pc-cygwin) libcurl/7.20.1 OpenSSL/0.9.8r zlib/
1.2.5 libidn/1.18 libssh2/1.2.5
> Host: demojam.com
```

```

> Accept: */*
> Authorization: OAuth oauth_consumer_key="J2iLG8a8xZPtqBIfHjIm",
oauth_nonce="_9CPbe4VwA",
    oauth_signature_method="HMAC-SHA1", oauth_timestamp="1316135032" ,
oauth_version="1.0",
    oauth_token="vbXkQoSkiQyYI2I2u", oauth_signature="7Jf
%2FLxrvWGRPzK0n7%2FQn5VrEYo4%3D"
> Content-Type: text/plain
>
< HTTP/1.1 200 OK
< Cache-Control: no-store, private
< Expires: -1
< Vary: *
< X-Runtime : 721
< ETag: "74b302e289e66afbbe423f35d6b6145b"
< Set-Cookie: _12sprints_srv=m13; path=/; secure; HttpOnly
< Set-Cookie: _cstar_session=15086b4712eef7278a51e13c91535825; path=/;
secure; HttpOnly
< Content-Type: text/html; charset=utf-8
< Content-Length: 40
< Date: Fri, 16 Sep 2011 01:04:01 GMT
< Connection: close
< Server: SAP LJS 1.0.0
< Set-Cookie: BIGipServerstage.demojam.com-443-pool=217718700.20480.0000;
path=/
<=====[ Response Below ]=====
controller=oauth_api&action=test_request

```

### 8.1.3.2.8 Revoke an Authorized Access Token

Use this procedure to have the client application revoke an authorized access token.

The rest API endpoint to do this is [POST] /oauth/revoke\_token, authorized as a regular 3-legged OAuth request.

It programmatically revokes the authorized access token that was used to authenticate the call, so that it can no longer be used for OAuth requests and will no longer appear in the list of authorized access tokens for the user.

Under normal circumstances, a client app will not need to use this endpoint. However, if the client app is able to create access tokens without user intervention, and the client app does not persist the authorized access token beyond the scope of its session, it is possible to flood the user's list of authorized access tokens that enable the user to initiate access token revocation. In such a case, it is best for the client app to revoke the access token.

Developer notes on the design:

- We took the precedent for our API authentication revocation design from **elance.com**.
- This is similar to the **revocation proposal for OAuth 2**, although written in a form consistent with the current API.
- Google supports **OAuth token revocation**, but in a very implementation dependent way.
- Finally, it is always possible to work around the absence of a programmatic API by calling the UI-based endpoint for token revocation, even though this is inconvenient. There is no additional security exposure introduced here.

For more information, see:

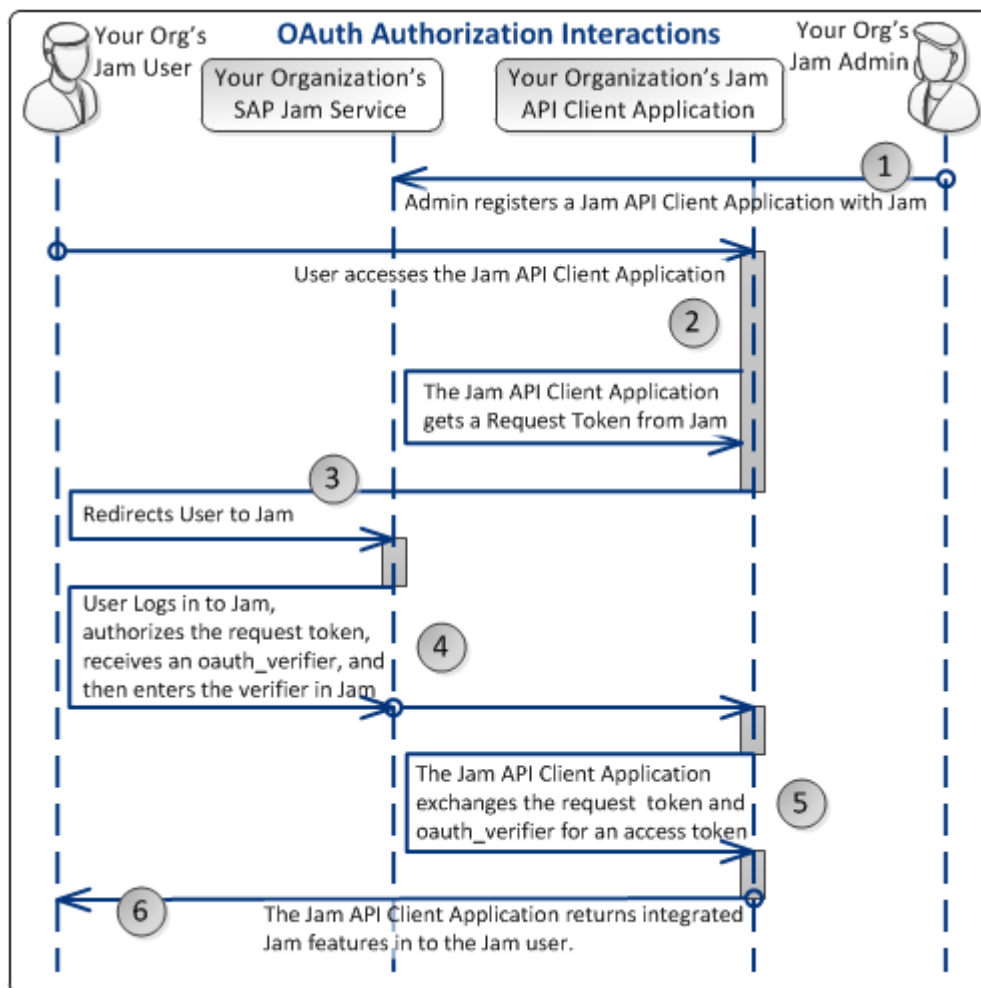
- <http://tools.ietf.org/html/draft-lodderstedt-oauth-revocation-03> ➡

- [https://developers.google.com/accounts/docs/OAuth\\_ref?csw=1#RevokeToken](https://developers.google.com/accounts/docs/OAuth_ref?csw=1#RevokeToken)

### 8.1.3.3 OAuth 1.0a Authorization API Reference

The API endpoints for the OAuth 1.0a workflow are documented in this section.

The API calls used in the workflow for this authorization method are as follows:



1. Your organization's SAP Jam Collaboration administrator registers your organization's SAP Jam Collaboration API client application with SAP Jam.
2. When the user views the SAP Jam API client application, it calls SAP Jam to obtain a request token (POST /oauth/request\_token).
3. The SAP Jam API client application then redirects the end user to SAP Jam.
4. The end user logs into SAP Jam and is presented with an authorization request to allow the SAP Jam API client application to make calls to SAP Jam on their behalf. If they accept, they receive an oauth\_verifier that they enter into SAP Jam Collaboration.
5. The SAP Jam API client application then exchanges the request token and the oauth\_verifier for an authorized access token (POST /oauth/access\_token).

- Subsequently, the SAP Jam Collaboration API client application can make public REST API calls on behalf of the end user, allowing the user to see SAP Jam elements in the SAP Jam Collaboration API client application.

The SAP Jam OAuth API also provides a utility call, `[GET] /oauth/test_request`, that allows you to validate an authorized access token.

It is also important to be able to revoke an authorized access token, which is done using `[POST] /oauth/revoke_token`.

### 8.1.3.3.1 [POST] /oauth/request\_token

Obtains a request token for the current user.

#### Request Elements

##### URL

`https://<jam#>.sapjam.com/oauth/request_token`

##### HTTP Request Method

POST

#### Authorization Header or Request Payload Parameters

Parameter	Required	Description
Type	True	The value must be "OAuth". This is actually OAuth 1.0a, an important security update of OAuth 1.0.
oauth_consumer_key	True	20 Base64 characters (120 bits). The authentication client "Consumer Key", obtained when the client app was registered with SAP Jam Collaboration.
oauth_nonce	True	A single-use randomly generated UUID string, used in all OAuth authentication requests and all subsequent authenticated API calls to prevent replay attacks. (Nonce = number, used once.)
oauth_signature	True	40 Base64 characters (240 bits). The encrypted hash of the canonical URL and parameters.
oauth_signature_method	True	The cryptographic method used for generating the signature. The valid values are PLAIN-TEXT, HMAC-SHA1, or RSA-SHA1.

Parameter	Required	Description
<b>oauth_timestamp</b>	True	The current UNIX timestamp.
<b>oauth_version</b>	True	1.0. The version of OAuth being used. This is actually OAuth 1.0a.
<b>oauth_callback</b>	True	<p>The Callback URL is the URL to which the user is returned after authorization.</p> <ul style="list-style-type: none"> <li>If this parameter value is <code>oob</code>, then the Callback URL that was provided when the client app was registered is used.</li> <li>If a Callback URL was provided when the client app was registered, but this parameter value is not <code>oob</code>, then the URL in the parameter must be within the same top-level domain as the provided URL. For example, <code>https://apis.example.com/callback</code> is valid, but <code>https://apis.somewhereelse.com</code> is not. The response code <code>401 Unauthorized</code> is displayed if an invalid URL is used.</li> </ul>

### Note

The above parameters can be supplied in the Authorization header or as query parameters in the POST request body, as specified in section 5.2 of the [OAuth spec](#) .

## Response Elements

If successful, the request returns HTTP 200 OK and a request token key and secret. The request token secret should be kept secret by the client app, and it should be stored until the user is prompted to authorize use of the token, which will require that the token be replaced with an OAuth authorization token.

## Response Properties

Property	Description
<code>oauth_token</code>	The returned OAuth 1.0a access token. This is the public part of the OAuth request token.
<code>oauth_token_secret</code>	The returned OAuth 1.0a auth token secret. This is the private part of the OAuth request token.
<code>oauth_callback_confirmed</code>	This is always "true".

### Note

The `request_token` is used to get an access token, but first the user must accept this, so the user must be redirected to SAP Jam Collaboration and presented with a prompt to accept this. The URL to which you must redirect the user is `https://demojam.com/oauth/authorize?oauth_token=<oauth_token>`. For more information, see the "OAuth 1.0a Authorization" section in the *Authorization API* documentation.



## Response Example

URL	Method
https://<jam#>.sapjam.com/oauth/request_token	POST

<pre>oauth_token=39g1abHfFJZgx7Z2lnos&amp;oauth_token_secret=A6bIuS15Bfs2qbH7G6D1d1UQ6FSgNP9J7G1gilEn&amp;   oauth_callback_confirmed=true</pre>
--------------------------------------------------------------------------------------------------------------------------------------------------

## HTTP Status Codes

HTTP Success Code	Description
200 OK	The requested resource retrieval is successful, and a full payload of the requested resource is returned.

HTTP Error Codes	Description
400 Bad Request	The server could not understand the request due to malformed syntax.
403 Forbidden	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
404 Not Found	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
405 Method Not Allowed	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
409 Conflict	There is an internal access conflict to the specified resource.
429 Too Many Requests	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
500 Internal Server Error	The server encountered an unexpected condition, which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.

### 8.1.3.3.2 [POST] /oauth/access\_token

Replaces a user's "request token" with an "access token" upon user authorization.

## Request Elements

### URL

`https://<jam#>.sapjam.com/oauth/access_token`


### HTTP Request Method

POST

### Authorization Header or Request Payload Parameters

Parameter	Required	Description
Type	True	The value must be "OAuth". This is actually OAuth 1.0a, an important security update of OAuth 1.0.
oauth_consumer_key	True	20 Base64 characters (120 bits). The authentication client "Consumer Key", obtained when the client app was registered with SAP Jam Collaboration.
oauth_nonce	True	A single-use randomly generated UUID string, used in all OAuth authentication requests and all subsequent authenticated API calls to prevent replay attacks. (Nonce = number, used once.)
oauth_signature	True	40 Base64 characters (240 bits). The encrypted hash of the canonical URL and parameters.
oauth_signature_method	True	The cryptographic method used for generating the signature. The valid values are PLAIN-TEXT, HMAC-SHA1, or RSA-SHA1.
oauth_timestamp	True	The current UNIX timestamp.
oauth_version	True	1.0. The version of OAuth being used. This is actually OAuth 1.0a.
oauth_token	True	The access token created using <a href="#">[POST] /oauth/request_token [page 419]</a> and returned to the user when they authorize use of the OAuth token. See <a href="#">2. Get a Request Token from Jam [page 411]</a> and <a href="#">4. User authorizes the client application [page 413]</a> for information on how this token is created and passed, respectively.
oauth_verifier	True	The oauth_verifier string that is returned to the user when they authorize use of the OAuth token. See <a href="#">4. User authorizes the client application [page 413]</a> for information on how this token is passed to the user.

#### Note

The above parameters can be supplied in the Authorization header or as query parameters in the POST request body, as specified in section 5.2 of the [OAuth spec](#) .

## Response Elements

If successful, the request returns an authorized access token key and secret. The access token secret should be kept secret by the client app. It should also be stored by the client app to avoid having to request end user authorization every time the user wants to access SAP Jam Collaboration resources.

## Response Properties

Property	Description
oauth_token	The returned OAuth 1.0a access token. This is the public part of the OAuth request token.
oauth_token_secret	The returned OAuth 1.0a auth token secret. This is the private part of the OAuth request token

## Response Example

URL	Method
https://<jam#>.sapjam.com/oauth/access_token	POST

```
oauth_token=kbXkRoS7IqyYIx8I2Ipu&oauth_token_secret=pK0z16l9GjgZddj7twsaD8ptWKQ48xrQi24xE2sA
```

## HTTP Status Codes

HTTP Success Code	Description
<b>201 Created</b>	The requested resource creation is successful, and a full payload of the created resource is returned.

HTTP Error Codes	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.

HTTP Error Codes	Description
<b>409 Conflict</b>	There is an internal access conflict to the specified resource.
<b>429 Too Many Requests</b>	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.

### 8.1.3.3.3 [GET] /oauth/test\_request

Checks if the access token is valid and working correctly without making unauthorized calls on behalf of the user.

## Request Elements

### URL

`https://<jam#>.sapjam.com/oauth/test_request`

## HTTP Request Method


GET

## Authorization Header or Request Payload Parameters

Parameter	Required	Description
<b>Type</b>	True	The value must be "OAuth". This is actually OAuth 1.0a, an important security update of OAuth 1.0.
<b>oauth_consumer_key</b>	True	20 Base64 characters (120 bits). The authentication client "Consumer Key", obtained when the client app was registered with SAP Jam Collaboration.
<b>oauth_nonce</b>	True	A single-use randomly generated UUID string, used in all OAuth authentication requests and all subsequent authenticated API calls to prevent replay attacks. (Nonce = number, used once.)

Parameter	Required	Description
<b>oauth_signature_method</b>	True	The cryptographic method used for generating the signature. The valid values are PLAIN-TEXT, HMAC-SHA1, or RSA-SHA1.
<b>oauth_timestamp</b>	True	The current UNIX timestamp.
<b>oauth_version</b>	True	1.0. The version of OAuth being used. This is actually OAuth 1.0a.
<b>oauth_token</b>	True	The access token created using <a href="#">[POST] /oauth/request_token [page 419]</a> and returned to the user when they authorize use of the OAuth token. See <a href="#">2. Get a Request Token from Jam [page 411]</a> and <a href="#">4. User authorizes the client application [page 413]</a> for information on how this token is created and passed, respectively.
<b>oauth_signature</b>	True	40 Base64 characters (240 bits). The encrypted hash of the canonical URL and parameters.

### Note

The above parameters can be supplied in the Authorization header or as query parameters in the POST request body, as specified in section 5.2 of the [OAuth spec](#) .

## Response Elements

## Response Properties

Property	Description
<code>controller</code>	Indicates the entity performing the action.
<code>action</code>	Indicates the operation that is being performed.

## Response Example

URL	Method
<code>https://&lt;jam#&gt;.sapjam.com/oauth/test_request</code>	GET
<code>controller=oauth_api&amp;action=test_request</code>	

## HTTP Status Codes

HTTP Success Code	Description
200 OK	The requested resource retrieval is successful, and a full payload of the requested resource is returned.

HTTP Error Codes	Description
400 Bad Request	The server could not understand the request due to malformed syntax.
403 Forbidden	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
404 Not Found	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
405 Method Not Allowed	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
409 Conflict	There is an internal access conflict to the specified resource.
429 Too Many Requests	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
500 Internal Server Error	The server encountered an unexpected condition, which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.

### 8.1.3.3.4 [POST] /oauth/revoke\_token

Revokes a token.

## Request Elements

### URL

`https://<jam#>.sapjam.com/oauth/revoke_token`

## HTTP Request Method

POST

## Authorization Header or Request Payload Parameters

Parameter	Required	Description
Type	True	The value must be "OAuth". This is actually OAuth 1.0a, an important security update of OAuth 1.0.
oauth_consumer_key	True	20 Base64 characters (120 bits). The authentication client "Consumer Key", obtained when the client app was registered with SAP Jam Collaboration.
oauth_nonce	True	A single-use randomly generated UUID string, used in all OAuth authentication requests and all subsequent authenticated API calls to prevent replay attacks. (Nonce = number, used once.)
oauth_signature	True	40 Base64 characters (240 bits). The encrypted hash of the canonical URL and parameters.
oauth_signature_method	True	The cryptographic method used for generating the signature. The valid values are PLAIN-TEXT, HMAC-SHA1, or RSA-SHA1.
oauth_timestamp	True	The current UNIX timestamp.
oauth_version	True	1.0. The version of OAuth being used. This is actually OAuth 1.0a.
oauth_token	True	The access token created using <a href="#">[POST] /oauth/request_token [page 419]</a> and returned to the user when they authorize use of the OAuth token. See <a href="#">2. Get a Request Token from Jam [page 411]</a> and <a href="#">4. User authorizes the client application [page 413]</a> for information on how this token is created and passed, respectively.
oauth_verifier	True	The oauth_verifier string that is returned to the user when they authorize use of the OAuth token. See <a href="#">4. User authorizes the client application [page 413]</a> for information on how this token is passed to the user.

### i Note

The above parameters can be supplied in the Authorization header or as query parameters in the POST request body, as specified in section 5.2 of the [OAuth spec](#) .

## Response Elements

### HTTP Status Codes

HTTP Success Code	Description
200 OK	The passed-in auth token has been revoked successfully.

HTTP Error Codes	Description
400 Bad Request	The server could not understand the request due to malformed syntax.

HTTP Error Codes	Description
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as <code>userId</code> , <code>groupId</code> , or <code>contentId</code> ).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal access conflict to the specified resource.
<b>429 Too Many Requests</b>	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.

## 8.1.3.4 SAML Authentication and Authorization

SAML API calls require a combination of OAuth and SAML to support the authentication and authorization work flow of SAP Jam Collaboration. This requires that both an OAuth client and a SAML Trusted IDP are registered by the SAP Jam company administrator.

Company administrators can create both OAuth clients and SAML IDPs via the [Admin](#) menu of SAP Jam. It is important to note that these OAuth clients can only be used for API calls authenticated by users who are members of the company for which the OAuth client is registered. This is also true for SAML assertions. In the case of SAML this feature is called a "Company Scoped IDP".

In this scenario SAP Jam acts as a SAML service provider (SP) with respect to the company-registered SAML IDP. These company-scoped SAML trusted IDPs are only used in various REST API flows. In particular, they do not support front-channel SSO or SLO. SAML assertions issued by these IDPs can only affect users within the company that the SAML trusted IDP is registered at.

### 8.1.3.4.1 SAML-based authentication

To implement SAML-based authentication you must create an OAuth2 access token from a SAML bearer assertion.

#### Create an OAuth2 access token from a SAML bearer assertion

```
[POST] /api/v1/auth/token
```



This API call is based on the <http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-14> specification:

- The parameter `client_id` is required with a value of the OAuth client consumer key.
- The parameter `client_secret` is required unless there is a `client_id` attribute in the SAML assertion. See below for details. We recommend not using the `client_secret` OAuth parameter and putting the `client_id` attribute in the SAML assertion, if possible.
- The parameter `grant_type` is required with the value `urn:ietf:params:oauth:grant-type:saml2-bearer`.
- The parameter `assertion` is required with the value of the SAML assertion.

#### i Note

The SAML assertion is not in a SAML unsolicited response envelope; it is just the assertion.

- The OAuth client and SAML trusted IDP must be configured for the same company. For example, you cannot use an OAuth client configured for Company A with a SAML trusted IDP configured for Company B.

For details on how the assertion is used to specify the user, see [Format Requirements for SAML Assertions and Unsolicited Responses \[page 429\]](#).

The <http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-14> specification states:

"Assertion authorization grants may be used with or without client authentication or identification. Whether or not client authentication is needed in conjunction with an assertion authorization grant, as well as the supported types of client authentication, are policy decisions at the discretion of the authorization server."

The SAP Jam Collaboration implementation requires client authentication. This ensures that the OAuth client for which the access token is granted is directly tied to the SAML assertion.

To do this:

- The `client_id` parameter is required. If a `client_secret` parameter is provided, it is checked. (This is required by the spec section 3.1: "If present, the authorization server MUST also validate the client credentials.")
- If a `client_id` attribute is provided in the assertion, then it must match the `client_id` OAuth parameter of the request. This ties the `client_id` to a signed value.
- If a `client_id` attribute is not provided in the assertion, then the `client_secret` parameter is required. This allows use with IDPs that have difficulties adding custom attributes (which can happen).

## 8.1.3.4.2 Format Requirements for SAML Assertions and Unsolicited Responses

This page describes the specific format requirements for SAML assertions and unsolicited responses.

The following are details on the requirements for a SAML unsolicited response or assertions used with the following authentication endpoints:

- `POST /api/v1/auth/token` (creates an authorized OAuth2 access token from a SAML2 assertion)

The specification of the user within SAP Jam Collaboration is determined *entirely* by data within the assertion (or unsolicited response). The Issuer element of the assertion must have the same value as the IDP ID for the SAML trusted IDP as configured by the company admin.

To identify the user in SAP Jam Collaboration by their email address, the SAML Subject NameID element can look like this:

```
<saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">anton.william@coke.com</saml:NameID>
```

The SuccessFactors user ID can also be used as the identifier for the user in SAP Jam. This is typically the user ID that the company "naturally" identifies its users with. The format is similar to this:

```
<saml:NameID Format="urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified" NameQualifier="www.successfactors.com">1819187</saml:NameID>
```

In fact, this format is superior to asserting based on email addresses, because email addresses are not guaranteed to be unique or even present within an SAP Jam company. For example, if two users share the same email, then an assertion based on an email address for one of those users will not work because it is ambiguous. Similarly, if a user doesn't have an email address, then an assertion based on email won't be able to specify that user, but an assertion based on the SuccessFactors userid will always work and uniquely identify the user.

To be specific:

- The Format attribute must be `urn:oasis:names:tc:SAML:1.1:nameid-format:unspecified`
- The NameQualifier attribute must be `www.successfactors.com`
- The NameID value must be the userid in the SuccessFactors IDP for the user

Note that SAML responses generated by the SuccessFactors platform IDPs use `www.successfactors.com` as their Issuer value within the Response. Thus, following the SAML standard, this NameQualifier value is required to let SAP Jam know that the NameID value is not being interpreted with respect to the native format of the IDP issuing the assertion, but rather with the SuccessFactors IDP.

In terms of other requirements on the assertion, there are additional checks on the audience, subject confirmation method, and recipient as required by OAuth2 in the SAML2 specification.

The audience should be `"cubetree.com"`. This is the main local SP name of SAP Jam and the recommended value. The SAML spec also allows the token endpoint URL of the authorization server to be used here (that is, the request URL without query parameters). The SuccessFactors IDP uses `"www.successfactors.com"` as the audience. We allow it for compatibility.

The subject confirmation method is required to be `urn:oasis:names:tc:SAML:2.0:cm:bearer`. The bearer subject confirmation is the only one that allows the Subject to be confirmed without further work on the part of the receiving SP.

The recipient is optional, for the sorts of assertions SAP Jam accepts, but if included it must match the base request URL. For example, `https://jam4.sapjam.com/v1/groups` for a [POST] `/v1/groups` from a SAML response call. SAP Jam checks that the scheme, host, port and path match, that is, it ignores query parameters.

Pretty much all assertions or SAML responses created by a standard IDP will include all these elements. They provide protection against the use of a valid assertion from an IDP being used at a SP that wasn't the intended target of the assertion, or in the context where the assertion was not meant to fully establish the identity of the user (for example, the bearer token).

## 8.1.3.5 SAML Authorization API Reference

The API endpoints for the SAML Authorization workflow are documented in this section.

### 8.1.3.5.1 [POST] /api/v1/auth/token

Create an OAuth2 access token from a SAML assertion.

#### Request Elements

##### URL

`https://<jam#>.sapjam.com/api/v1/auth/token`

##### HTTP Request Method

POST

##### HTTP Request Header

Field	Required	Description
Content-Type	Required	The Content-Type header must be <code>application/x-www-form-urlencoded</code> .

##### POST Body Parameters

Field	Required	Description
client_id	Required	The OAuth client consumer key.
client_secret	Not Recommended	The client_secret is only required if there is not a client_id attribute in the SAML assertion. We recommend that you do not use the client_secret parameter, and that you do use the client_id attribute.

Field	Required	Description
<b>grant_type</b>	Required	The grant_type value must be <code>urn:ietf:params:oauth:grant-type:saml2-bearer</code> . <b>However</b> , each of the colons must be set in URL encoding as '%3A', so, the actual string is: <code>urn%3Aietf%3Aparams%3Aoauth%3Agrant-type%3Asaml2-bearer</code>
<b>assertion</b>	Required	The SAML authorization assertion of the identity of the user for whom you want an auth token. This must be Base64 encoded.

### Note

For more details, please see the SAML Bearer Assertion specification, <http://tools.ietf.org/html/draft-ietf-oauth-saml2-bearer-14>.

## Response Elements

### JSON Response Example

URL	Method
<code>https://&lt;jam#&gt;.sapjam.com/api/v1/auth/token?\$format=json</code>	POST

```
{
  "access_token": "<!-- [An Auth token has been removed from here.] -->"
}
```

## HTTP Status Codes

HTTP Success Code	Description
<b>200 OK</b>	The requested auth token is retrieved successfully.

HTTP Error Codes	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as <code>userId</code> , <code>groupId</code> , or <code>contentId</code> ).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal access conflict to the specified resource.

HTTP Error Codes	Description
<b>429 Too Many Requests</b>	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.

#### i Note

To test the retrieved auth token, use it in a simple GET operation, such as the `[GET] /Groups ('{Id}')` API call.

## 8.1.3.6 Authentication with single-use tokens

A single-use token can be used to authenticate a user web session that is valid for only one request to SAP Jam. You can create a new single-use token by making an API call to SAP Jam using a `[POST] /v1/single_use_tokens` HTTP request. Every API call has a user context/authorization. The token will be used to create a web session for the authenticated API user at the time of the creation of the token. A successful request returns a token that is used to authenticate a SAP Jam web session.

### 8.1.3.6.1 [POST] /v1/single\_use\_tokens

A single-use token is typically a very short-lived token that is used to display some content without requiring the end-user to log in to view it or without providing the user with a longer lived `auth_token` that can be used multiple times. This page shows you how to create a single-use token.

## Request Elements

### URL

`https://{jam_domain}/v1/single_use_tokens`

### HTTP Request Method

POST

## Response Elements

### XML Response Example

URL	Method
https://{jam_domain}/v1/single_use_tokens	POST

<pre>&lt;?xml version="1.0" encoding="UTF-8"?&gt; &lt;single_use_token id="&lt;!--[The single-use token has been removed from here.]--&gt;"&gt;   &lt;created_at&gt;2012-12-15T03:03:08Z&lt;/created_at&gt;   &lt;expires_by&gt;2012-12-15T04:03:08Z&lt;/expires_by&gt; &lt;/single_use_token&gt;</pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

As the name implies, once you have retrieved the token, it is good for only one use. It also is typically given a very short lifespan. It is used by appending it to a URL that would normally require a log in.

### HTTP Status Codes

HTTP Success Code	Description
<b>201 Created</b>	The requested resource creation is successful, and a full payload of the created resource is returned.

HTTP Error Codes	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal access conflict to the specified resource.
<b>429 Too Many Requests</b>	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.

## 8.1.3.6.2 Test the single\_use\_token

This page demonstrates the use of a single use token.

### Request Elements

Use a single-use token to authenticate a user to allow them to access a specific secure endpoint for a single instance of access. It is used by appending the single-use token to a URL that requires authentication to access.

### URL

The following URL is to a SAP Jam page that requires authentication to view. By appending a single-use token, a user can access that page one time, and they are not provided with a token that will allow them access to other pages or endpoints. The generated single-use token can only be used once, and it typically is given a very short lifespan.

```
https://{jam_domain}/groups/dashboard?single_use_token=_<single_use_token>
```

### HTTP Request Method

GET

### HTTP Input Headers

Header Field	Use	Description
Authorization: Bearer <access_token>	Mandatory	The Authorization header field should be set as shown, with "<access_token>" replaced with the current user's access token.

### Response Elements

The response contains a lot of HTML to render the page, which is omitted here.

## HTTP Status Codes

HTTP Success Code	Description
200 OK	The requested resource retrieval is successful, and a full payload of the requested resource is returned.
<hr/>	
HTTP Error Codes	Description
400 Bad Request	The server could not understand the request due to malformed syntax.
403 Forbidden	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
404 Not Found	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (such as userId, groupId, or contentId).
405 Method Not Allowed	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
409 Conflict	There is an internal access conflict to the specified resource.
429 Too Many Requests	The user has sent too many requests in a given period of time. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.
500 Internal Server Error	The server encountered an unexpected condition, which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.

### 8.1.4 SAP Jam REST API

The SAP Jam Collaboration REST API has mostly been ported to the OData API; however, one set of REST API calls remains:

- The REST Social Reports API calls allow you to generate and retrieve reports on social activity in SAP Jam.

#### 8.1.4.1 REST Social Reports API Calls

The Social Reports SAP Jam Collaboration Search API calls allow you to generate and retrieve reports on the social activity in SAP Jam Collaboration.



## 8.1.4.1.1 [GET] /reports

Returns a list of administrator reports.

### i Note

[Administrator reports](#) can only be generated, viewed and downloaded by company administrators.

## URL

`https://<jam#>.sapjam.com/api/v1/reports`

## HTTP Request Method

GET

## HTTP Input Headers

Header Field	Use	Description
<b>Authorization: Bearer &lt;access_token&gt;</b>	Required	The Authorization header field should be set as shown, with <access_token> replaced with the current user's access token.

## JSON Response Example

URL	Method
<code>https://&lt;jam#&gt;.sapjam.com/api/v1/reports</code>	GET

```
[
  {
    "id":44,
    "type":"activity_summary_by_week",
    "title": "(=Activity Summary Report ((=weekly=)))",
    "submitted":"2013/10/02 23:49:14 +0000",
    "status":"done",
    "completed":"2013/10/02 23:49:25 +0000",
    "identifier":"9917a240-704d-4fb9-b4fc-dd36d29d5d99"
  },
  {
    "id":45,
    "type":"usage_auditing",
    "title":"Content Viewers Report",
    "submitted":"2013/10/02 23:49:26 +0000",
    "status":"done",
    "completed":"2013/10/02 23:49:30 +0000",
    "identifier":"15165908-9389-4189-b4cb-e37a2c4638cb"
  }
]
```

## Error Codes

HTTP Error Code	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (userId, groupId, contentId, etc.).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal conflict of access to the specified resource.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.
<b>509 Bandwidth Limit Exceeded</b>	The server is temporarily unable to service your request. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.

### 8.1.4.1.2 [POST] /reports

Generates a new administrator report which displays information on a specified aspect of SAP Jam usage.

#### i Note

[Administrator reports](#) can only be generated, viewed and downloaded by company administrators.

## URL

`https://<jam#>.sapjam.com/api/v1/reports`

## HTTP Request Method

POST

## HTTP Input Headers

Header Field	Use	Description
Authorization: Bearer <access_token>	Required	The Authorization header field should be set as shown, with <access_token> replaced with the current user's access token.

## Input Parameters

Field	Type	Required	Description
type	string	Required	The type of report to generate. The supported types are listed in the "Available Reports" table below.

## Available Reports

Report	Required Parameters	Group-level Access	Features
activity_summary_by_week	start_date, end_date	yes	
activity_summary_by_month	start_date, end_date	yes	
company_user_detail_report			
content_consumption_by_week	start_date, end_date	yes	
content_consumption_by_month	start_date, end_date	yes	
disk_usage_per_group			disk_usage_info

Report	Required Parameters	Group-level Access	Features
disk_usage_per_user			disk_usage_info, user_level_reporting
engagement_report			
group_activity_report			
group_member_activity_report		yes	user_level_reporting
kudo_detail	start_date, end_date		kudos
user_activity_report	start_date, end_date	yes	user_level_reporting
user_contribution_report_by_week	start_date, end_date	yes	user_level_reporting
user_contribution_report_by_month	start_date, end_date	yes	user_level_reporting
contribution_by_object_report_by_week	start_date, end_date	yes	
contribution_by_object_report_by_month	start_date, end_date	yes	
user_consumption_by_week	start_date, end_date	yes	user_level_reporting
user_consumption_by_month	start_date, end_date	yes	user_level_reporting
idea_report	start_date, end_date	Group only	ideas
usage_auditing	date		user_level_reporting

## JSON Response Example 1 - company\_user\_detail\_report

URL	Method
<a href="https://&lt;jam#&gt;.sapjam.com/api/v1/reports?type=company_user_detail_report">https://&lt;jam#&gt;.sapjam.com/api/v1/reports?type=company_user_detail_report</a>	POST

```
{
  "id":218,
  "type":"company_user_detail_report",
  "title": "(=User Details Report=)",
  "submitted":"2013/10/23 18:01:43 +0000",
  "status":"scheduled",
  "completed":nil,
  "identifier":nil
}
```

## JSON Response Example 2 - group\_member\_activity\_report

URL	Method
https://<jam#>.sapjam.com/api/v1/reports? type=group_member_activity_report&group_id=K3cgYl17t01zEwOUUIbUb5	POST

```
{
  "id": 678,
  "type": "group_member_activity_report",
  "title": "Group Member Activity Report",
  "submitted": "2018/06/08 11:48:55",
  "status": "scheduled",
  "completed": null,
  "identifier": null
}
```

## Error Codes

HTTP Error Code	Description
400 Bad Request	The server could not understand the request due to malformed syntax.
403 Forbidden	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
404 Not Found	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (userId, groupId, contentId, etc.).
405 Method Not Allowed	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
409 Conflict	There is an internal conflict of access to the specified resource.
500 Internal Server Error	The server encountered an unexpected condition, which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.
509 Bandwidth Limit Exceeded	The server is temporarily unable to service your request. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.

### 8.1.4.1.3 [GET] /reports/<ID>

Gets metadata for the administrator report specified by the ID in the request URL.

#### i Note

[Administrator reports](#) can only be generated, viewed and downloaded by company administrators.

## URL

`https://<jam#>.sapjam.com/api/v1/reports/<ID>`

## HTTP Request Method

GET

## HTTP Input Headers

Header Field	Use	Description
<b>Authorization: Bearer &lt;access_token&gt;</b>	Re-quired	The Authorization header field should be set as shown, with <access_token> replaced with the current user's access token.

## JSON Response Example

URL	Method
<code>https://&lt;jam#&gt;.sapjam.com/api/v1/reports/&lt;ID&gt;</code>	GET

```
{
  "id":218,
  "type":"company_user_detail_report",
  "title": "(=User Details Report=)",
  "submitted":"2013/10/23 18:01:43 +0000",
  "status":"scheduled",
  "completed":nil,
  "identifier":nil
}
```

## Error Codes

HTTP Error Code	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.

HTTP Error Code	Description
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (userId, groupId, contentId, etc.).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal conflict of access to the specified resource.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.
<b>509 Bandwidth Limit Exceeded</b>	The server is temporarily unable to service your request. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.

### 8.1.4.1.4 [GET] /reports/<ID>/content

Downloads the content of the administrator report specified by the ID in the request URL.

#### i Note

[Administrator reports](#) can only be generated, viewed and downloaded by company administrators.

## URL

`https://<jam#>.sapjam.com/api/v1/reports/<ID>/content`

## HTTP Request Method

GET

## HTTP Input Headers

Header Field	Use	Description
<b>Authorization: Bearer &lt;access_token&gt;</b>	Required	The Authorization header field should be set as shown, with <access_token> replaced with the current user's access token.

## JSON Response Example

URL	Method
https://<jam#>.sapjam.com/api/v1/reports/<ID>/content	GET

<pre>(=User ID=), (=First Name=), (=Last Name=), (=Title=), (=Email=), (=Country/Region=), (=Joined?=), (=First Login At=),     (=Disabled At=), (=Last Login At=), (=Admin?=), (=Status=), (=Invited By=) ,,, abc@cubetree.com, (=No=), "", "", "", (=No=), (=Pending=), Deepthi Sunder ,,, asdf@cubetree.com, (=No=), "", "", "", (=No=), (=Pending=), claireR155 ChiAr155 ,,, asdfasdfasdfasdfasdfasdfasdf@cubetree.com, (=No=), "", "", "", (=No=), (=Pending=), claireR155 ChiAr155 ,,, asdfasf@cubetree.com, (=No=), "", "", "", (=No=), (=Pending=), claireR155 ChiAr155</pre>
---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

## Error Codes

HTTP Error Code	Description
400 Bad Request	The server could not understand the request due to malformed syntax.
403 Forbidden	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
404 Not Found	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (userId, groupId, contentId, etc.).
405 Method Not Allowed	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
409 Conflict	There is an internal conflict of access to the specified resource.
500 Internal Server Error	The server encountered an unexpected condition, which prevented it from fulfilling the request.
501 Not Implemented	The server does not support the functionality required to fulfill the request.
509 Bandwidth Limit Exceeded	The server is temporarily unable to service your request. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.

### i Note

The download is the content of the report. In most cases, this is the form of a CSV file. In the case of the "usage\_auditing" report, this is a ZIP file.



### 8.1.4.1.5 [GET] /reports/<ID>/is\_available

Used to check if the administrator report specified by the ID in the request URL has been processed.

#### i Note

[Administrator reports](#) can only be generated, viewed and downloaded by company administrators.

## URL

`https://<jam#>.sapjam.com/api/v1/reports/<ID>/is_available`

## HTTP Request Method

GET

## HTTP Input Headers

Header Field	Use	Description
<b>Authorization: Bearer &lt;access_token&gt;</b>	Required	The Authorization header field should be set as shown, with <access_token> replaced with the current user's access token.

## JSON Response Example

URL	Method
<code>https://&lt;jam#&gt;.sapjam.com/api/v1/reports/&lt;ID&gt;/is_available</code>	GET

```
{
  "is_available":false
}
```

## Error Codes

HTTP Error Code	Description
<b>400 Bad Request</b>	The server could not understand the request due to malformed syntax.
<b>403 Forbidden</b>	Access to the resource you are trying to connect to is forbidden. This may be due to either a user authentication failure or to the user having insufficient privileges to perform the action.
<b>404 Not Found</b>	The server cannot find the specified resource. This is typically due to an unrecognized resource ID (userId, groupId, contentId, etc.).
<b>405 Method Not Allowed</b>	The method specified in the Request Line is not allowed for the resource identified by the Request URI.
<b>409 Conflict</b>	There is an internal conflict of access to the specified resource.
<b>500 Internal Server Error</b>	The server encountered an unexpected condition, which prevented it from fulfilling the request.
<b>501 Not Implemented</b>	The server does not support the functionality required to fulfill the request.
<b>509 Bandwidth Limit Exceeded</b>	The server is temporarily unable to service your request. This error message is typically encountered if API rate limits, set to protect against DoS attacks and to preserve server responsiveness, have been exceeded.

## 9 SAP BTP

### i Note

As of November 2020, the SAP Jam developer edition has been deprecated and you can no longer access a free trial instance of SAP Jam from your SAP BTP account.

SAP is introducing a new digital workplace solution, which includes the enterprise social networking features known from SAP Jam Collaboration: SAP Work Zone And SAP Work Zone for HR.

There is currently no test instance of SAP Work Zone available on the SAP BTP. To develop for SAP Work Zone, you have to buy a license as a regular customer.

SAP Jam Collaboration is still fully supported and maintained. SAP offers a smooth transition path for SAP Jam customers to bring content to the new software.



For more information about SAP Work Zone, see [SAP Work Zone](#) on the SAP Help Portal.

# 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 a 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.



© 2022 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.