<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.1</td>
<td>Access to Security Relevant Attributes</td>
<td>32</td>
</tr>
<tr>
<td>7.2</td>
<td>Support of Binary Attributes</td>
<td>33</td>
</tr>
<tr>
<td>7.3</td>
<td>Search Entries</td>
<td>33</td>
</tr>
<tr>
<td>7.4</td>
<td>Get Specific Entry</td>
<td>38</td>
</tr>
<tr>
<td>7.5</td>
<td>Update an Entry</td>
<td>43</td>
</tr>
<tr>
<td>7.6</td>
<td>Get Attribute Values of an Entry</td>
<td>50</td>
</tr>
<tr>
<td>7.7</td>
<td>Add Attribute Values to an Entry</td>
<td>54</td>
</tr>
<tr>
<td>7.8</td>
<td>Update Attribute Values of an Entry</td>
<td>58</td>
</tr>
<tr>
<td>7.9</td>
<td>Search Entry References</td>
<td>60</td>
</tr>
<tr>
<td>7.10</td>
<td>Context Usage</td>
<td>64</td>
</tr>
<tr>
<td>7.11</td>
<td>Virus Scanning</td>
<td>64</td>
</tr>
<tr>
<td>8</td>
<td>Task Access URIs</td>
<td>66</td>
</tr>
<tr>
<td>8.1</td>
<td>Get All Self-Service Tasks</td>
<td>66</td>
</tr>
<tr>
<td>8.2</td>
<td>Get a Specific Self-Service Task</td>
<td>68</td>
</tr>
<tr>
<td>8.3</td>
<td>Get All Task Attributes</td>
<td>70</td>
</tr>
<tr>
<td>8.4</td>
<td>Get a Specific Task Attribute</td>
<td>73</td>
</tr>
<tr>
<td>8.5</td>
<td>Get Allowed Values of a Specific Task Attribute</td>
<td>76</td>
</tr>
<tr>
<td>9</td>
<td>Approval Access URIs</td>
<td>78</td>
</tr>
<tr>
<td>9.1</td>
<td>Get Open Approval Requests for Current User</td>
<td>78</td>
</tr>
<tr>
<td>9.2</td>
<td>Process an Approval</td>
<td>81</td>
</tr>
<tr>
<td></td>
<td>Get Valid Decision Options for Approval</td>
<td>81</td>
</tr>
<tr>
<td></td>
<td>Execute Approval Decision</td>
<td>82</td>
</tr>
<tr>
<td>10</td>
<td>Attestation Access URIs</td>
<td>85</td>
</tr>
<tr>
<td>10.1</td>
<td>Get Attestation Requests for Current User</td>
<td>85</td>
</tr>
<tr>
<td>10.2</td>
<td>Process an Attestation</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td>Get Valid Decision Options for Attestation</td>
<td>89</td>
</tr>
<tr>
<td></td>
<td>Execute Attestation Decision</td>
<td>91</td>
</tr>
<tr>
<td>10.3</td>
<td>Get a History of Attestations</td>
<td>93</td>
</tr>
<tr>
<td>11</td>
<td>Service Operations</td>
<td>97</td>
</tr>
<tr>
<td>11.1</td>
<td>Get Version Information</td>
<td>97</td>
</tr>
</tbody>
</table>
1 SAP Identity Management REST Interface Version 2

Version 2 of Identity Management REST (Representational state transfer) service implements the Open Data Protocol (OData) in version 2.0.

Consumers of this service API (Application Programming Interface) should be familiar with OData before implementing their own applications. For more information about the Open Data protocol, see the OData web site.

Version 2 of the Identity Management REST service supports - as does OData - both formats for representing the resources it exposes, the XML-based Atom format and the JavaScript Object Notation (JSON) format. In contrast, version 1 of the Identity Management REST service only supported the JSON format. For readability’s sake, all examples in this document are based on the Content-Type application/json, with the exception of the service metadata document that is only returned in XML format.

OData is designed to be modular. This means, that the Identity Management REST service only implements as much of the OData specification as required for the intended target scenarios and used the odata4j open source library. Furthermore, the support for Query Options is limited in version 2. To allow the implementation of further features, for example, to support more Query Options, within the REST interface in later service packs, clients can call a service operation that returns version information. Clients can then check whether certain features can be used or not.

The Identity Management REST interface supports SAP NetWeaver 7.3 SP9 and 7.31 SP6 and higher as runtime environment and requires SAP NetWeaver Identity Management 7.2 SP8 or higher. The interface is contained in the IDMREST<IdM SP version>_<IdM Patch version>.sca.

It is still possible to use the version 1 of the Identity Management REST service. However, the Identity Management REST Interface Version 2 has clear advantages over its previous version, and is the recommended version of the Identity Management REST service.

Related Information

Open Data Protocol

1.1 Security

This section covers the following security topics for the REST Interface:

- Authentication and authorization
  - Necessary UME actions
  - Better performance with single sign-on with logon tickets
1.1.1 Authentication and Authorization

Most calls to the provided service URIs (Uniform Resource Identifiers) require a user to authenticate. The user must have the UME actions `idmAuthenticated` and the `idm_authenticated_restapi`.

As the REST service is an application deployed on the SAP NetWeaver AS for Java, you can also use logon ticket authentication instead of basic authentication to approve performance.

Related Information

Configuring the UME Actions for the REST Interface [page 25]
Configuring Single Sign-On With Logon Tickets in the REST Interface for AS Java 7.1 and higher [page 26]

1.1.1.1 Single Sign-On With Logon Tickets

The default configuration of the SAP NetWeaver Identity Management 7.2 REST interface forces a logon on all requests using the provided basic authentication credentials. This authentication consumes time and leads to a high number of security sessions in the SAP NetWeaver AS for Java.

The system has to maintain these sessions until they time out. This impacts the performance of the REST interface for subsequent requests.

To improve performance, configure single sign-on (SSO) with logon tickets for the REST interface.

Related Information

Configuring Single Sign-On With Logon Tickets in the REST Interface for AS Java 7.1 and higher [page 26]

1.1.2 Cross-Site Request Forgery Protection

The Identity Management REST service protects against Cross-Site Request Forgery (CXRF/XSRF) attacks. Cross-Site Request Forgery is an attack that tricks a victim’s browser into sending a request to a vulnerable web application, which then performs an undesired action on behalf of the victim (for example, changing credentials, making an illegal purchase, or performing online financial operation). This vulnerability is extremely widespread since the majority of today’s web applications rely solely on automatically submitted credentials such as session
cookies, basic authentication credentials, source IP addresses, Secure Socket Layer (SSL) certificates, or Microsoft Windows domain credentials. Therefore, as the user is currently authenticated to the site, the site will have no way to distinguish the XSRF attack from a legitimate user request.

To protect against attacks of this type, the Identity Management REST interface uses a standard filter provided by the SAP NetWeaver Application Server for Java (AS Java).

- Enable the Virus Scan Interface to prevent XSRF attacks.
- The XSRF protection for the SAP NetWeaver REST interface is based on an XSRF token and applies to all modifying requests. The protection works as follows:
  - With a non-modifying request, authenticate to the server and retrieve an XSRF token. The client sends a non-modifying request (GET, HEAD, OPTIONS), where the X-CSRF-Token header field has the value Fetch. This request also includes the regular authentication information required by the REST interface, like user ID and password, logon ticket. The SAP SAP NetWeaver Identity Management REST interface generates an XSRF token and sends it back in the X-CSRF-Token header of the response. This token is associated with the session established between the client and the server.

  **Note**

  The response of a failed request does not contain a valid XSRF token.

  - With each modifying request, include the XSRF token. The client sends a modifying request, where the same request header (X-CSRF-Token) has the value of the previously received token. The server validates the XSRF token. Additionally, the request includes authentication data (as far as required again, depending on the login module stack used for the REST interface) and especially all cookies set by the server during the non-modifying request. The cookies include session information, which is required by the server to verify the XSRF token because of its association with the current session. If the validation fails, the server replies with an error response with HTTP status code 403 (Forbidden).

  **Note**

  If two services are in the same security policy domain, authenticating with the first service allows the user to access the second one without additional authentication. Also, the received XSRF token is valid for all services across the security policy domain.

**Related Information**

- Authentication and Authorization [page 5]
- XSRF Protection for REST Services
- Specifying Authentication Mechanisms for Java Applications (Description for setting the security policy domain.)
- SAP Identity Management Identity Center REST API
- Virus Scanning [page 64]
1.2 Content Type Negotiation

The REST service can use the existing HTTP headers to indicate or negotiate the representation of the content type of a request or response. Additionally, it allows the usage of an OData System Query Option to request a content type of a response.

1.2.1 Content Type Via Request Header

The REST client can specify which representation of the resource it prefers by including the HTTP request header “accept” with a list of MIME types it can handle.

If the header is not specified, the Atom representation is returned by default. If the requested MIME type is not supported by the REST service, the error code 406 (Not Acceptable) is returned. When a resource is sent as part of the request or response body, the client and server have to indicate the corresponding content type in the header content-type.

**Example**

HTTP GET Request

Accept: application/json

**Example**

HTTP POST Request

Accept: application/json

Content-Type: application/json

1.2.2 Content Type Via System Query Option

Using the Format System Query Option ($format), the REST client can specify which representation of the resource the response returns. If a request URI contains this option, it overrides the value specified in the accept request header.

Syntax

formatQueryOp = "$format=" ("json" / "atom")

**Example**

HTTP Request

GET ./idmrestapi/v2/service?$format=json HTTP/1.1
1.2.3 Character Encoding

The expected character encoding when sending requests to the REST interface is UTF-8.

The used character encoding is specified in the HTTP response header `Content-Type`. Currently this is always UTF-8.

Example

HTTP Request

```
Charset: UTF-8
```

Example

HTTP Response

```
Content-Type: application/json; charset=UTF-8
```

1.3 Localization

The REST interface supports the retrieval of localized texts for which translated texts exist. This applies, for example, to display names of attributes or localized value help texts contained in attributes.

If the locale of the corresponding step is not specified or not maintained in Identity Management, the locale is determined in the following order. This order is similar to the one described for the user management engine (UME) in User Profile: “If a language is not explicitly defined for a user in the user profile, then the browser language is used. If the browser language is not set, then the server’s default language is used.”.

Order for determining the language:

1. The language of the logged-in user.
2. The REST client can send the preferred language by sending the `Accept-Language` HTTP request header.
3. The default locale of the Java Virtual Machine (JVM).
4. English is used as the default language.

The used language is specified in the HTTP response header `Content-Language`.

Example

HTTP Request

```
Accept-Language: en
```

Example

HTTP Response
1.4 Performance

Context

1.4.1 OData Means

Following the OData specification, the REST service by default does not return the full graph of entries related to an entry or collection of entries requested by a URI.

It is deferred unless the client explicitly asks for this information using the \$expand System Query Option for these related entries. This conserves resources if this information is not needed, but also allows getting all information about related entries in one request if it is required by the client.

**Tip**

If a scenario always requires additional information that is deferred by default, we recommend that you use the \$expand option (if available), instead of always retrieving the information with additional REST calls to the "deferred content".

**Note**

If too many entries exist, retrieving a collection of entries (for example, executing a search for entries) may return a partial result list. Partial result lists are marked with a \_next property that contains the link to the next partial set of entries.

Related Information

Expand System Query Option
1.4.2 Response Properties

According to the OData standard the REST service by default always delivers all properties defined in the Service Metadata Document for an entity at all operations that return an entity or entity set.

As the amount of properties for an entry type usually is quite large, the HTTP response is also large. Consequently, the response time is rather long. But most clients are not interested in all these properties, especially as only for the properties that are exposed by the specified task values are returned. For all other properties the initial values are set.

Therefore, the Identity Management OData Service provides a custom query option. Using this option delivers only those single value attributes in a response that contain a value and are exposed by the specified task. This custom query option works for all operations that retrieve an entity or entity set.

If in addition other multi-value or entry reference attributes should be returned, they have to be specified with the system query option $select.

Request Parameters

Parameter = 'selectSetSVAttributes' [ 'true' ]

Request Example 1

Return all single value attributes of the person with the given ID that are exposed by the given task and contain a value.

GET/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?selectSetSVAttributes

Request Example 2

Return all single value attributes of the person with the given ID that are exposed by the given task and contain a value. In addition, also return the multi-value attribute MX_ADDITIONAL_PHONE.

GET/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?selectSetSVAttributes&$select=MV_MX_ADDITIONAL_PHONE
1.5 Comparison to the Previous Version of the Identity Management REST Interface

Version 2 of the Identity Management REST service has clear advantages over its previous version, and is the recommended version of the Identity Management REST service.

It is still possible to use the version 1 of the Identity Management REST service, but you should consider upgrading to SAP Identity Management REST Interface Version 2.

**Note**
Upgrading to the Identity Management REST Interface Version 2 from a previous version will involve a reconfiguration (reimplementation) of your existing solution.

1.5.1 URIs

The URIs for the different entities have changed slightly as compared to the first version of the REST interface.

For example, accessing an element of a list used to be done like this:

```
list/<element1>
```

With version 2 of the interface an element is accessed by putting the value in brackets:

```
list(<element1>)
```

All items of the list are accessed in the same way as in the first version, for example:

```
list
```

**Note**
In accordance with the OData specification all URIs in version 2 of the OData Rest interface are case-sensitive.

**Example Version 1**

```
http://<host>:<port>/idmrest/v72alpha/entries/190/tasks/
```

**Example Version 2**

```
```
1.5.1.1 Operations

The second version of the REST interface supports all HTTP methods that are supported by the OData service interface, except the MERGE HTTP method. As a limitation of the used odata4j library a MERGE request is not supported.

Instead, method tunneling must be done via POST HTTP method including the header "X-HTTP-Method" with the value MERGE. The HTTP methods have a uniform meaning over all the entities it can handle.

Table 1: Supported HTTP Methods

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve an entity or a set of entities</td>
<td>GET</td>
</tr>
<tr>
<td>Create a new entity</td>
<td>POST</td>
</tr>
</tbody>
</table>

1.5.2 Operations

The first version of the REST interface only supports the HTTP methods GET and POST for managing all supported operations (retrieve, create and update) on the exposed entries.

Version 2 of the REST interface supports all HTTP methods that are supported by the OData service interface, except the MERGE HTTP method. As a limitation of the used odata4j library a MERGE request is not supported. Instead, method tunneling must be done via POST HTTP method including the header "X-HTTP-Method" with the value MERGE.

The HTTP methods have a uniform meaning over all the entities it can handle.

Table 2: HTTP Methods for Operations

<table>
<thead>
<tr>
<th>Operation</th>
<th>HTTP Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieve an entity or a set of entities</td>
<td>GET</td>
</tr>
<tr>
<td>Create a new entity</td>
<td>POST</td>
</tr>
<tr>
<td>Update an entity (via merge)</td>
<td>POST with MERGE method tunneling</td>
</tr>
<tr>
<td>Delete an entity</td>
<td>DELETE</td>
</tr>
</tbody>
</table>

1.5.3 Return Format

While the return format of the first interface version was JSON-only, version 2 of the REST interface also supports the Atom format.

The JSON response and the data structure within version 2 of the REST interface are adapted to be OData compliant. "In responses payloads (not request payloads) all the JSON representations […] are wrapped by an
outer most object that includes a single name/value pair. The name of the name/value pair is always "d" and the
value is the JSON representation of an OData resource [...]. This pattern ensures JSON payloads returned from
OData services are valid JSON statements, but not valid JavaScript statements. This prevents an OData JSON
response from being executed as the result of a cross site scripting (XSS) attack.”

Example Version 1

```json
{
    "MX_REST_SUCCESS":true,
    "ENTRIES":{
        "MX_MAIL_PRIMARY":{
            "DISPLAYVALUE":"max.mustermann@example.com",
            "VALUE":"max.mustermann@example.com"
        },
        "MX_FIRSTNAME":{
            "DISPLAYVALUE":"Max",
            "VALUE":"Max"
        },
        "MX_PHONE_PRIMARY":{
            "DISPLAYVALUE":"+49 6227 8-12345",
            "VALUE":"+49 6227 8-12345"
        },
        "MX_MOBILE_PRIMARY":{
            "DISPLAYVALUE":"+49 123 4567890",
            "VALUE":"+49 123 4567890"
        },
        "DISPLAYNAME":{
            "DISPLAYVALUE":"Max Mustermann",
            "VALUE":"Max Mustermann"
        },
        "MX_LASTNAME":{
            "DISPLAYVALUE":"Mustermann",
            "VALUE":"Mustermann"
        },
        "MSKEYVALUE":{
            "DISPLAYVALUE":"D0123456",
            "VALUE":"D0123456"
        }
    }
}
```

Example Version 2

```json
{
    "d":{
        "metadata":{
            "__uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=190,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
            "type":"IdMRestModel.ET_MX_PERSON"
        },
        "ID":190,
        "TASK_GUID":"35B264BF-A75A-447D-A4EA-7894725245CE",
        "SV_MSKEYVALUE":"D0123456",
        "SV_DISPLAYNAME":"Max Mustermann",
        "SV_MX_LASTNAME":"Mustermann",
        "SV_MX_FIRSTNAME":"Max",
```
Related Information

JSON Representations
Atom Representations

1.5.4 Query Options

With the second version of the Identity Management REST interface new features like "expanding" are introduced. For an overview of possible operators, see the OData web site.

Related Information

Query String Options

1.6 Identity Store Support

Version 2 of the REST interface only supports the identity store specified in the Identity Management JMX application property com.sap.idm.jmx.idstoreid, as did the first version of the REST interface.

Related Information

Installing and Configuring the Identity Management User Interface
1.7 Limitations

This overview lists the limitations of the second version of the Identity Management REST service.

- Response Header
  - The response header does not contain the Content-Language in the HTTP response header.
  - The response header returns `DataServiceVersion: 1.0` although the returned response complies with version 2.0.
- If an error occurs, the error response always shows the language “en-US”, even if the error text is localized.
- Attributes with the data type “Audit flag”, “Task reference”, and “Attribute reference” are not supported and are not returned in an HTTP response.
- Paging of search results is not supported yet by the Identity Management Rest Service. For more information, see Search Entries.
- The return value for attributes with non-text data types and the “hide input” presentation property is always null. For more information, see Get Attribute Values of an Entry.
- Currently, it is not possible to create new entities using the REST interface.

Related Information

Get Attribute Values of an Entry [page 50]
Search Entries [page 33]

1.8 Compatibility

As the REST interface advances and different versions of the specification materialize, we need to define how the SAP Identity Management REST interface provider (provider, for short) and the SAP Identity Management REST interface consumers (consumers, for short) are supposed to interoperate. This section defines the compatibility rules that need to be adhered to.

Version Numbering

To properly communicate the idea of compatible and incompatible changes, compatible version increments MUST only be minor versions, not major versions. On the other hand, incompatible changes MUST require a new major version. As far as this document is concerned, the goal is to only extend compatibly, hence introducing only minor versions. Features added through minor versions but the first non-beta version of a major release MUST be optional.

Each incompatible change to the OData service is reflected in a different major version. The version is embodied in the URI, leading to a new service endpoint with each such change. If exceptions are required, for example, for security reasons, they are explicitly communicated.
Rule 1: Introduction of Compatible Changes

Unless otherwise noted, each newer SAP Identity Management REST interface version is fully compatible with respect to previous versions. If a compatible change is introduced, a new minor version is required. This minor version number increment is reflected by the function import `GetVersion`, see Get Version Information.

The following changes are compatible (this list is very restrictive and is likely to be extended in the future):

- Adding new function imports (actions and functions), association sets, and entity sets to existing entity containers.
- Defining new associations, entity types, and complex types.
- Adding new properties or navigation properties to existing entity types if they meet the following requirements:
  - The properties are optional (nullable).
  - The navigation properties are optional.
- Adding new optional parameters to existing function imports, given that calling the new function import without the new parameter has the same meaning as calling the old function import for any given value combination for the existing parameters, for example, \( f_{\text{new}}(x, \text{null}) = f_{\text{old}}(x) \).
- Defining new optional custom query options.
- Introducing paging support. Clients shall be prepared to support partial sets of entities as defined in the OData specification, see the OData web page.

It is important that consumers honor these rules appropriately. That means, consumers MUST expect the above changes to happen and MUST NOT rely on any technical means that would assume otherwise, for example, static proxy generation (a well-known flaw in the SOAP web service world) to consume content.

Rule 2: Introduction of Incompatible Changes

All changes not listed under rule 1 are incompatible.

Rule 3: Feature Querying and Default Values

Even for fully compatible evolutions, a consumer must know the features the provider supports. For major version changes a new endpoint and service URI are introduced. If the consumer is newer than the provider, it could, for example, try to invoke an action that is not implemented. To remedy this situation, consumers must be able to query the features the provider supports. OData service metadata documents provide the right mechanism for this. Taking this technical detail as a basis, we follow the convention that the SAP Identity Management Rest interface exposes the features supported in the service metadata document. Additionally the consumer can call the `GetVersion` function import to retrieve the major and minor version to find out if a required feature is available.
Rule 4: Provider Backward Compatibility

The versioning scheme employed defines that the provider (which is bound to a particular endpoint and supports a specific minor version of a major version) is compatible to all lower minor versions of this major version. Beyond that, the provider is not forced to remain backward compatible. Specifically, new major versions are typically implemented in a separate provider bound to a separate endpoint.

Rule 5: Provider Forward Compatibility

The provider is not required to be forward compatible. That is, the provider is not required to be able to talk to consumers supporting only newer major versions of the SAP Identity Management REST interface. Within the boundaries of a major release, except for a set of (initially) defined mandatory features, new features added through minor versions MUST be optional, hence making it possible for the provider supporting a lower minor version of a specific major version to communicate with a consumer that supports a higher minor version of that same major version.

Rule 6: Consumer Backwards Compatibility

Consumers do not have to be backward compatible. That is, they do not have to support all (latest minor of all major) SAP Identity Management Rest interface versions – they can choose to support only one. Within the boundaries of a major version, however, the consumer SHOULD be able to deal with providers of a lower version (at the cost of a moderately reduced feature set).

Rule 7: Consumer Forward Compatibility

As outlined in rule 1, a consumer MUST be able to distinguish those pieces of a provider’s response that it needs and disregard all those it does not understand. Note this notion of forward compatibility only applies within the boundaries of compatible versions of the SAP Identity Management REST interface. So as long as a consumer understands the SAP Identity Management REST interface 2.0 and the provider offers 2.1, everything should be fine. If the provider only offers 3.0, the consumer has no chance.

Consumers are advised to make use of the $select query option to avoid unnecessary data passing between provider and consumer.

Related Information

Get Version Information [page 97]
OData web page
2 Deploying the REST Interface Version 2

Prerequisites

Before you can deploy the SAP Identity Management REST Interface Version 2, make sure that the following prerequisites are fulfilled:

- One of the following SAP NetWeaver versions must be correctly installed and licensed:
  - SAP NetWeaver 7.3 SP9 Patch 1 or higher.
  - Enhancement Package 1 for SAP NetWeaver 7.3 SP6 Patch 3 or higher.
- SAP OData library (odata4j) is required. The required library is available as an AS Java Extension for SAP NetWeaver version you are using (versions 7.3 SP9 Patch 1 and higher, or EHP1 for SAP NetWeaver 7.3 SP6 Patch 3 and higher). Download the library extension from the SAP Software Download Center and deploy the downloaded SCA file on your AS Java server, using the Software Update Manager (SUM).

  **Note**
  Follow the following path on the SAP Software Download Center to locate the correct OData library:
  
  Support Packages and Patches > A - Z Index > N > SAP NETWEAVER > <your SAP NETWEAVER version> > Entry by Component > AS Java Extensions > SAP ODATA4J+CXF-REST LIB <your SAP NW version> > # OS independent

- SAP Identity Management Identity Center 7.2 SP8 or higher must be correctly installed and licensed.
- SAP Identity Management User Interface 7.2 SP8 or higher needs to be correctly installed and configured.

Context

When all prerequisites are fulfilled, you can proceed with deploying the REST Interface Version 2. To deploy the REST Interface Version 2, do the following:

Procedure

1. Start by downloading the SCA file for the REST component which is to be deployed. Navigate to the download area of SAP NetWeaver Identity Management 7.2 on SAP Support Portal and the SAP Software Download Center, and download the SCA file.

  **Note**
  Follow the following path on the SAP Software Download Center to locate the correct SCA file for the REST component:
  
  Support Packages and Patches > A - Z Index > N > SAP NW IDENTITY MANAGEMENT
Note

Make sure that the SCA file for the REST Interface Version 2 has the same SP version as the SAP Identity Management and its User Interface. The SCA file name is IDMREST<IdM SP version>_IdM Patch version>.sca. E.g. for SAP NetWeaver Identity Management 7.2 SP8 (Patch 0), the file name is IDMREST08_0.sca.

2. Use Software Update Manager (SUM) to deploy the SCA file on your SAP NetWeaver AS for Java where the SAP Identity Management User Interface is deployed.

Related Information

- SAP Software Download Center
  Installation Information for SAP NetWeaver 7.3/Enhancement Package 1 for SAP NetWeaver 7.3
  SAP Identity Management Identity Center Installation overview
  SAP Identity Management Identity Center Installing and configuring the Identity Management User Interface

2.1 Using the Software Update Manager (SUM)

The Software Update Manager (SUM) is a multi-purpose tool that supports various processes, such as performing a release upgrade, installing enhancement packages, applying Support Package Stacks, installing add-ons, or updating single components on SAP NetWeaver.

Prerequisites

- Make sure that the Software Update Manager is downloaded and available on your SAP NetWeaver AS Java. The SUM is part of the Software Logistics Toolset delivery and available for download from SAP Software Download Center. Support Packages and Patches. A - Z. Index. S. SL TOOLSET. SL TOOLSET 1.0. Entry by Component. SOFTWARE UPDATE MANAGER (SUM).

- You can find the documentation for SUM on the SL Toolset page on SAP Service Marketplace. For SAP NetWeaver AS for Java, there are specific guides for the combinations of operating systems and databases.

Note

To find the documentation describing the SUM, from the Software Logistics Toolset page on SAP Service Marketplace, navigate to Software Logistics Toolset 1.0. System Maintenance. Document. System Management at the bottom of the page.
Before running and using the SUM, you have to complete all required preparation and planning actions in the SUM user guide.

Make sure that the SAP system and its database are started.

On the host where you want to start the SL Common GUI of the Software Update Manager, Java 6 or higher has to be installed.

SAP Host Agent has been configured on your system with the minimum version required for your scenario. For more information, see Installing or Updating SAP Host Agent in the Update of SAP Systems Using Software Update Manager guide that is relevant for your operating system and database.

**Context**

To start and use the Software Update Manager, proceed as follows:

**Procedure**

1. Run the Software Update Manager on the application server of the primary application server instance.
2. Start the SL Common GUI of the Software Update Manager.
3. Logon to the Software Update Manager and deploy the SCA file.

**Related Information**

SAP Software Download Center
SL Toolset page on SAP Service Marketplace
Running the Software Update Manager [page 21]
Starting the SL Common GUI of the Software Update Manager [page 22]
Deploying Using the Software Update Manager [page 23]
2.1.1 Running the Software Update Manager

Context

To run the Software Update Manager on the application server (primary application server instance), proceed as follows:

Procedure

1. Log on to the host on which the primary application server instance is running as user <SAPSID>adm (instance user).
2. Unpack the Software Update Manager package (<archive>.SAR) with the following command:
   - for Microsoft Windows:
     SAPCAR -xf <download directory>/<path>/<Archive>.SAR -R <DRIVE>:\usr\sap\<sapsid>
     This command creates the directory SUM under the <DRIVE>:\usr\sap\<sapsid> directory. You can also specify a directory other than <DRIVE>:\usr\sap\<sapsid>. In the following, the directory \<path to SUM directory>\SUM is referred to as <update directory>.
   - for UNIX:
     SAPCAR -xf <download directory>/<path>/<Archive>.SAR -R /usr/sap/<sid>
     This command creates the directory SUM under the /usr/sap/<sid> directory. You can also specify a directory other than /usr/sap/<sid>. In the following, the directory /<path to SUM directory>/SUM is referred to as <update directory>.
   - Note
     The complete path to the SUM folder must not exceed 30 characters.
3. Start the Software Update Manager entering the following command:
   - for Microsoft Windows:
     <DRIVE>:\<update directory>\STARTUP.BAT confighostagent
     For Microsoft Windows and MS SQL Server, enter the following command:
     <DRIVE>:\<update directory>\STARTUP.BAT confighostagent jvm6
   - for UNIX:
     cd /usr/sap/<SID>/SUM
     ./STARTUP confighostagent <SID>
Related Information

Using the Software Update Manager (SUM) [page 19]

2.1.2 Starting the SL Common GUI of the Software Update Manager

Context

This section describes how you start the SL Common UI and the SUM back-end process.

Procedure

1. Open a web browser window.
2. In the address bar, enter the following URL: https://<hostname>:1129/lmsl/sumjava/<SID>/index.html.
   Replace <hostname> with the name of the host on which the Software Update Manager is running.
   
   Note
   
   If the SSL is not configured, use http instead of https at the beginning of the URL, and use port 1128: http://<hostname>:1128/lmsl/sumjava/<SID>/index.html.

3. In the dialog box that appears, enter the user name <sid>adm and the password.

Results

The SAP Host Agent starts the Software Update Manager, and the SL Common GUI of the Software Update Manager is displayed in the web browser.
Related Information

Using the Software Update Manager (SUM) [page 19]

2.1.3 Deploying Using the Software Update Manager

Context

The Software Update Manager controls the entire procedure, from checking the system requirements and importing the necessary programs through stopping production operation until production operation is resumed. The procedure is divided up into a number of different roadmap steps. The roadmap steps are in turn divided into phases. Many phases require no user input - step through those by choosing Next. The successful completion of a phase is a precondition for the success of all subsequent phases.

Note

User actions are also required when errors occur. If an error occurs, correct it and repeat the phase in which the error has occurred. Once the phase has been repeated successfully, you can continue with the update.

To logon to the Software Update Manager and deploy the SCA file(s), do the following:

Procedure

1. Enter the user name and the password for the AS Java Administrator user with which you log in to the system.
2. In the Specify Credentials roadmap step, specify the password for the instance user (<sapsid>adm), and then choose Next.
3. In the Select Target roadmap step, specify the path to the SCA file in the Directory field, then choose Next.
4. In the Confirm Target roadmap step, enter the keyword that is specified in the current Central Software Update Manager Note (which you can find in the Software Update Manager upgrade guide or in SAP Support Portal). Confirm the selected target system version by choosing Next.
5. In the Configuration roadmap step, provide the password of the AS Java Administrator before proceeding. In this step it is also possible to specify the composition of the target release system.
6. Step through the phases requiring no user input by choosing Next and complete the process. Upon completing the process successfully, the important statistics are collected in a comprehensive report.
Next Steps

Every time you have used SUM, you need to either delete the SUM folder or rename it and keep it (if you would like, but this is not necessary). Then you have to extract a new SUM folder from the SUM.SAR file.

Use SAPCAR.EXE to extract the SAR file. Do the following:

1. In the command prompt, change to the directory to which you have downloaded or copied the SUM archives (the directory of the SUM.SAR file).
2. Start SAPCAR to extract the archive to the current directory. Enter
   `<path to sapcar.exe>`\sapcar.exe -xvf SUM.SAR
   and run the command line.
3. The SUM.SAR file should now be extracted and the new SUM folder created. You may now use SUM again.

Related Information

SAP Notes on SAP Support Portal
Using the Software Update Manager (SUM) [page 19]
3 Configuring the UME Actions for the REST Interface

The necessary UME actions `idm_authenticated` and the `idm_authenticated_restapi` are provided as part of the software components containing the Identity Management web user interface and the REST service. To use the REST service and to access the Service Document and Service Metadata Document the user needs these UME actions.

In the UME administration, use the search term `tc~idm*` to find these actions. All other necessary authorizations for a service call are defined by the access control of the related Identity Management UI task.

Related Information

- Administration of Users and Roles in User Management Engine (UME) for SAP NetWeaver 7.3
- Administration of Users and Roles in User Management Engine (UME) for SAP NetWeaver 7.3 EHP1
- Authentication and Authorization [page 5]
4 Configuring Single Sign-On With Logon Tickets in the REST Interface for AS Java 7.1 and higher

Context

Using SSO with logon tickets, each request sent to the REST interface without a logon ticket creates a logon ticket. The example application in SAP Note 1665777 shows how the logon tickets are read from the HTTP response and how they are attached to a subsequent HTTP request for the same user.

Note

The logon ticket contains authentication information of the user of the first request (without logon ticket). If you use multiple users for calls to the REST interface, you have to ensure that the tickets are only used for the correct user.

If the application calling the REST interface is an application deployed on an SAP NetWeaver AS for Java, for example, a portal application, you may want to reuse the logon ticket already issued for the logged-on user.

The following code snippet shows how an existing logon ticket of the logged-on user is received from the UME interface and forwarded with the HTTP request:

```java
IUser loggedInUser = UMFactory.getAuthenticator().getLoggedInUser();
if (loggedInUser != null) {
    String sapLogonTicket = (String) loggedInUser.getTransientAttribute(  
        IPrincipal.DEFAULT_NAMESPACE,  
        ILoginConstants.SSOTICKET_USER_ATTRIBUTE_PURE);
    Map cookieStore = new HashMap();
    cookieStore.put("MYSAPSSO2", sapLogonTicket);
    //enrich HTTP request with cookies as shown in the class HTTPClient  
    //method setCookies(...) in the example.zip attached to SAP Note 1665777
}
```

Note

If the REST client application and the SAP NetWeaver Identity Management 7.2 REST interface reside on different servers and you want to use existing logon tickets issued by the server running the REST client application, you need to configure a trust relationship for logon tickets between the two servers. For more information about how to configure this relationship with SAP NetWeaver AS Java based systems, see Using Logon Tickets in the SAP Library that applies to your release.

To enable authentication for systems based on SAP NetWeaver AS for Java 7.1 and higher using logon tickets, you must configure the SAP Identity Management 7.2 REST Interface. The interface texts may vary slightly between releases.
Procedure

1. Log on to SAP NetWeaver Administrator at http(s)://<server>:<port>/nwa.
   For AS Java 7.31, choose Configuration Management ➔ Security ➔ Authentication.
3. Enter sap.com/tc~idm~rest~ear in the Policy Configuration Name filter and set the Type filter to Web.
4. Select sap.com/tc~idm~rest~ear.
5. Change to edit mode using the Edit button.
6. Choose the ticket authentication template.
7. Save the changes using the Save button.
   For AS Java 7.31, choose Home ➔ Operations ➔ Systems ➔ Start & Stop.
9. Choose the Java Applications tab.
10. In the application list, filter for tc~idm~rest~ear.
11. Select the tc~idm~rest~ear application.
12. Stop the application using the Stop button ➔ On All Instances.
13. Start the application using the Start button ➔ On All Instances.

Related Information

SAP Note 1665777 - SSO Configuration of SAP NetWeaver ID Mgmt 7.2 REST API Using Logon Tickets (This is the 7.3 EhP1 topic.)
Single Sign-On With Logon Tickets [page 5]
Authentication and Authorization [page 5]
5 Upgrading the REST Interface Version 2

Context

To perform an upgrade of an already deployed Identity Management REST Interface Version 2 component, do the following:

**Note**

SCA file for the Identity Management REST Interface Version 2 must be on the same SP level as SAP Identity Management (and its User Interface). Upgrading the Identity Management REST Interface Version 2 to a new SP version requires upgrade of the other component to the same SP version first.

Procedure

Update the Identity Management REST Interface Version 2 by deploying the new SCA file as described in *Deploying the REST Interface Version 2*.

It is not necessary to update the configuration after update of the REST Interface Version 2 component.

Related Information

[Deploying the REST Interface Version 2](#)
6 General Service URIs

Version 2 of the Identity Management REST interface provides two types of metadata documents to describe the service, the Service Document and the Service Metadata Document.

6.1 Service Document

The service document is available at the Service Root URI. It returns a list of all available top-level collections. It does not return a list of the available service operations, like getrequests. These can be found in the Service Metadata Document.

**URI**

(*) [GET] *

URI = './idmrestapi/v2/service';

**Request Parameters**

None.

**Request Example**

GET /idmrestapi/v2/service HTTP/1.1

**Response Example**

{ "d" : {
   ]
}}
6.2 Service Metadata Document

The Service Metadata Document describes the data model of the Identity Management OData service. It describes the available entity types, the relationships among them and the available service operations. For more information, see the OData web page. The service metadata is returned as XML format only as defined by the OData specification.

URI

(* [GET] *)
URI = './idmrestapi/v2/service/$metadata';

Request Parameters

None.

Request Example

GET /idmrestapi/v2/service/$metadata HTTP/1.1

Related Information

http://www.odata.org
7 Entry Access URIs

The following URIs allow accessing Identity Store entries, for example, Persons and Roles. You can query for entries, access a certain entry or access and update entry attributes. You can use tasks with the *This task creates a new entry* flag set, only for `POST` (create) operations. Otherwise an error code and message are returned. You can use tasks, for which the flag is not set, for read and modify operations.

Entry Attributes

The entry attributes, for example, `MSKEYVALUE`, and `DISPLAYNAME`, which are returned in any entry response, are defined on the entry type. All attributes of the entry type are marked *Allow*.

However, the attribute values are only set for those attributes contained in the specified task of the respective entry URI. To avoid empty attributes in an entry response, you can use custom query options in search entries or get entry requests.

Attribute Prefixes

Due to the underlying data model and to avoid name conflicts, the attribute names of an entry type are not returned 1:1 in the entry response. Every attribute gets a prefix depending on the type of the attribute.
Table 3: Attribute Prefixes

<table>
<thead>
<tr>
<th>Attribute Type</th>
<th>Prefix</th>
<th>Example for Identity Management</th>
<th>Example in REST Interface</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single Value Attribute</td>
<td>SV_</td>
<td>MX_FIRSTNAME</td>
<td>SV_MX_FIRSTNAME</td>
</tr>
<tr>
<td>Multi-Value Attribute</td>
<td>MV_</td>
<td>MX_MAIL_ADDITIONAL</td>
<td>MV_MX_MAIL_ADDITIONAL</td>
</tr>
<tr>
<td>Entry Reference Attribute</td>
<td>ER_</td>
<td>MXREF_MX_ROLE</td>
<td>ER_MXREF_MX_ROLE</td>
</tr>
</tbody>
</table>

In the system query options of the entry requests, use these prefixes as well when an attribute name is specified, for example, in $filter and $select.

7.1 Access to Security Relevant Attributes

Read and write access to entries require specific handling as compared to accessing task meta information.

The information in this section applies to all read and write accesses for entries.

Returning Encrypted or Hashed Attribute Values

The return values for attributes differ depending on the `hide input` presentation property defined on the task attribute and on whether the connection is secured using SSL or not.

- `hide input` flag is enabled
  - `{HIDDEN}` is returned for non-encrypted, encrypted and hashed attributes.
- `hide input` flag is disabled
  - If the attribute is encrypted and SSL is used, the decrypted value is returned.
  - If the attribute is hashed and SSL is used, the value (hash) is returned.
  - If the attribute is encrypted or hashed but no SSL is used, `{HASHED}` or `{ENCRYPTED}` is returned.

If the attribute does not have a value, `null` is returned.

Encrypted Attributes

To set values for attributes with enabled encryption flag, you must use SSL. Otherwise, the value is not stored and an error with a fitting message is returned to the client. If you use SSL, the JMX layer encrypts the value specified in the HTTP request. The value is then stored in the identity store.
Hashed Attributes

To set values for hashed attributes, you must use SSL. Otherwise, the value is not stored and an error with a fitting message is returned to the client. If you use SSL, the JMX layer encrypts the value specified in the HTTP request. The value is then stored in the identity store.

Setting the Password for a User

To set the password for a user, you must use SSL, and the client should set the value using the \texttt{MX\_PASSWORD} attribute. The value is hashed and stored for the \texttt{MX\_PASSWORD} attribute. If the option “Enable password provisioning” is set on the Identity Store, the value is encrypted and stored for the \texttt{MX\_ENCRYPTED\_PASSWORD} attribute. This is the same behavior as in the web enabled tasks.

\textbf{i \ Note}

If the value for the \texttt{MX\_ENCRYPTED\_PASSWORD} attribute is specified, the value is encrypted and stored for the \texttt{MX\_ENCRYPTED\_PASSWORD} attribute but no value for the \texttt{MX\_PASSWORD} attribute is set.

7.2 \ Support of Binary Attributes

The REST interface supports binary data like pictures. These attributes are of type \texttt{File} defined in the \texttt{metadata.xml} document. The \texttt{File} type is a complex data type, consisting of meta information about the binary attribute value.

The \texttt{File} type has a special \texttt{SIZE} pseudo property that contains the size of the media object in bytes.

To access the actual binary value of a binary attribute, append \texttt{$value} to the URI of the \texttt{File} entry. The complete URI is also contained in the property \texttt{media\_src} of the \texttt{File} entry.

7.3 \ Search Entries

Searches for identity store entries of the specified entry type name (\texttt{MX\_ENTRYTYPE}).

It supports one optional OData parameter (\texttt{$filter}) where a search expression with search attributes can be specified. And the custom query option (\texttt{filterBasic}) to specify the search filter for the basic search. Also a simple combined search is possible by using both parameters. The search result is a collection of entries.

In the current version 2, the REST interface supports only the “and” operator.
Search Attributes

The attributes that can be used in the search depend on the definition of a search and/or display task for the respective entry type. The determination order is the following:

1. If a search task is defined for the entry type, any attribute contained in this task can be used.
2. If only a display task is defined for the entry type, any attribute contained in this task can be used.
3. If neither a search nor display task is set, any attribute that is selected in the List column of the entry type can be used.
4. If also no attribute in the List column is selected, the default search attribute is the MSKEYVALUE.

Search Result

The attribute values of an entry returned in the search result are the attributes selected in the List column of the corresponding entry type. In addition to those, the MSKEY is returned for every entry as this is a key (as defined in the metadata document).

Note

Every response must have a task GUID, because it is a part of the key of an entry. However, the returned attribute values of the search result are not determined by a task, but by the attributes marked as “List” on the entry type. Therefore, the special task GUID FFFFFFFF-FFFF-FFFF-FFFFFFFFFFFF is used in a search response that indicates this behavior.

Maximum Search Results

The maximum amount of entries returned depends on the Max search result size property, which is set on the General tab of the Identity Store in the Identity Center. If a search result exceeds this number, the maximum amount of entries is returned in the response. And by default the response will contain a “__next” property with a link that contains the OData query option $skiptoken with the value searchLimitExceeded. This indicates
that the search result may not be complete. If a client follows this link, it only gets a 404 - Not Found response, because paging of partial search results is not implemented yet by the Identity Management Rest Service. To deactivate this behavior and to not show the "__next" property in the search result, set the application property v2.ShowNextLinkWhenSearchLimitIsExceeded to false.

1. In the SAP NetWeaver Administrator, choose Configuration > Infrastructure > Java System Properties.
2. On the Applications tab page, filter for tc-idm-rest-ear.
3. On the Properties tab page, select v2.ShowNextLinkWhenSearchLimitIsExceeded, choose Modify, and enter false in the Enter Custom Value field.

**URI**

```{shell}
(* [GET] *)
GET URI                         = './idmrestapi/v2/service/ET_ EntryTypeName;
EntryTypeName               = Name of an entry type (as specified in attribute
MX_ENTRYTYPE of each entry) that exists in the Identity Store schema;
```

**Request Parameters**

**Note**

The URI has to be URL escaped to replace special characters, like the space with "%20".

**Status Search**

```{shell}
Parameter                 = ['?' BasicFilter]
BasicFilter   = 'filterBasic=' SearchValue;
SearchValue    = Edm.String; (* * is wildcard *)
```

With this filter parameter an OR search is performed for all search attributes (see above).

**Advanced Search**

```{shell}
Parameter                             = '?' Filter
Filter                 = '$filter=' SearchAttributes;
SearchAttributes      = KeyValuePair (Junction KeyValuePair)*;
Junction              = ' and '; (* case sensitive! *)
KeyValuePair          = FilterAttributeName Comparator AttributeValue;
FilterAttributeName   = AttributePrefix AttributeName
AttributePrefix       = 'SV_' | 'MV_' (* single- or multi-value attribute *
*)
Comparator                  = Equal
```
Search Operators

The REST interface supports in the first version only the "and" operator.

Allowed Attribute Names

- The attribute names allowed in the search filter are the ones from the search attributes (see above).
- If a specified attribute name is not contained in the search attributes, the request is invalid. This results in a 400 - Bad Request response.

Attribute with Value Help

- For attributes that have a value help, the value key has to be specified as search value. A search for the value text is not supported.

Combined Search

- The combined search logically combines the results for both searches with an "AND" operator.
- Only one search attribute is supported for the advanced search filter.

Request Example 1

Basic search for all entries of entry type MX_PERSON.

GET /idmrestapi/v2/service/ET_MX_PERSON

Request Example 2

Advanced search for entries of entry type MX_PERSON with the first name "Mirco*" and the last name "Schuster*".

GET /idmrestapi/v2/service/ET_MX_PERSON?$filter=SV_MX_FIRSTNAME eq 'Mirco*' and SV_MX_LASTNAME eq 'Schuster*'

Request Example 3

Basic search for entries of entry type MX_PERSON with the value "Mirco" in one of the search attributes.

GET /idmrestapi/v2/service/ET_MX_PERSON?filterBasic=Mirco
Request Example 4

Combined search for entries of entry type MX_PERSON with the value “Mirco” in one of the search attributes and the department “Sales”.

GET /idmrestapi/v2/service/ET_MX_PERSON?filterBasic=Mirco&$filter=SV_MX_DEPARTMENT eq 'Sales'

Response Types

Content-Type: application/json | application/atom+xml

HTTP Response Codes

- 200 OK
  Response Body: Collection of entities with the type of the requested entry type (MX_ENTRYTYPE).
- 400 Bad Request
  ○ If the $filter parameter does not comply with the rules defined in section Parameters or if the attribute name is not an attribute of the corresponding entry type.
  ○ If the way the $filter and filterBasic parameters are used together is not supported.
- 403 Forbidden
  Logged-in user is not allowed to search for entries.
- 404 Not Found
  If the specified entry type name does not exist.

Response Example 1

```json
{"d":{
  "results": [{
    "__metadata": {
      "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=4321,TASK_GUID= guid'FFFFFFFF-FFFF-FFFF-FFFFFFFFFFFF')",
      "type": "MX_PERSON",
    },
    "ID": 4321,
    "SV_MSKEYVALUE": "mircoschuster1",
    "SV_DISPLAYNAME": "Mirco Schuster",
    "SV_MX_FIRSTNAME": "Mirco",
    "SV_MX_LASTNAME": "Schuster",
  },
  {
    "__metadata": {
      "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=4322,TASK_GUID= guid'FFFFFFFF-FFFF-FFFF-FFFFFFFFFFFF')",
    }
  }
}}
```
7.4 Get Specific Entry

Returns a specific entry for a given MSKEY containing a set of its attributes that are defined by the passed task and are set for the entry.

The TaskGUID must originate from a UI task and the currently logged-in user must be able to access this task.

The type of an attribute depends on whether it is a single-value, a multi-value or an entry reference attribute. For more information, see Get Attribute Values of an Entry.

For the single-value attributes that do not contain a value, the Null value is returned.

**URI**

```java
(* [GET]*)
URI     = './idmrestapi/v2/service/ET_' EntryType '(ID=' Mskey
       ' TaskGUID= ' TaskGUID ')

EntryType = Edm.String;
Mskey     = Edm.Int32;
TaskGUID  = Edm.Guid;
```
Note
The IdStoreID is not required here, because the MSKEY is unique across all Identity Stores.

Request Parameters

Supported Query Options:

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$expand</td>
<td>To inline deferred multi-value or entry reference attributes.</td>
</tr>
<tr>
<td>$select</td>
<td>To only return a subset of the attributes that are exposed by the specified task.</td>
</tr>
</tbody>
</table>

Request Example 1

GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE') HTTP/1.1

Request Example 2

GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?$expand=ER_MXREF_MX_ROLE HTTP/1.1

Request Example 3

GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?$expand=MV_MX_PHONE_ADDITIONAL HTTP/1.1

Response Types

Content-Type: application/json
HTTP Status Codes:

- **200 OK**
  Response Body: Entry

- **403 Forbidden**
  - Logged-in user is not allowed to access the task.
  - Task is a self-service task and the MSKEY is not from the logged-in user.

- **404 Not Found**
  If the specified entry does not exist.

Response Example 1

```json
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "MX_PERSON"
      },
      "ID": 7894,
      "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
      "SV_MSKEYVALUE": "schuessler1",
      "SV_MX_LASTNAME": "Schüssler",
      "SV_MX_FIRSTNAME": "Mo",
      "SV_MX_MAIL_PRIMARY": "mo.schuessler@example.com",
      "SV_DISPLAYNAME": "Mo Schüssler",
      "MV_MX_PHONE_ADDITIONAL": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/MV_MX_PHONE_ADDITIONAL"
        }
      },
      "ER_MXREF_MX_ROLE": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/ER_MXREF_MX_ROLE"
        }
      }
    }
  }
}
```

**Note**

For more information about attributes that contain binary data (for example, BI_MX_USER_PICTURE, which can contain pictures with content types like image or jpeg), see Get Attribute Values of an Entry.
Response Example 2

```json
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "MX_PERSON"
      },
      "ID": 7894,
      "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
      "SV_MSKEYVALUE": "schuessler1",
      "SV_MX_LASTNAME": "Mo",
      "SV_MX_FIRSTNAME": "Schuessler",
      "SV_MX_MAIL_PRIMARY": "mo.schuessler@example.com",
      "SV_DISPLAYNAME": "Mo Schüssler",
      "MV_MX_PHONE_ADDITIONAL": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/MV_MX_PHONE_ADDITIONAL"
        }
      },
      "ER_MXREF_MX_ROLE": {
        "__metadata": {
          "uri": "/idmrestapi/v2/service/ER_MXREF_MX_ROLE(LINK_ID=1737,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
          "type": "EntryReference"
        },
        "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
        "LINK_ID": 1737,
        "REFERENCED_ID": 41203,
        "REFERENCED_DISPLAY_NAME": "role 3",
        "VALID_FROM": "/Date(1360454400000)/",
        "VALID_TO": "/Date(1362095999000)/",
        "REASON": "undefined - test",
        "STATE": 4,
        "STATE_TEXT": "Pending",
        "DIRECTLY_ASSIGNED": true,
        "INHERITED": false,
        "OPERATION": null,
        "CONTEXT": null,
        "REFERENCED_ENTRY": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1737,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/REFERENCED_ENTRY"
          }
        },
        "CONTEXT_ENTRY": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1737,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/CONTEXT_ENTRY"
          }
        }
      }
    },
    "__metadata": {
      "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
      "type": "MX_PERSON"
    },
    "ID": 7894,
    "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
    "SV_MSKEYVALUE": "schuessler1",
    "SV_MX_LASTNAME": "Mo",
    "SV_MX_FIRSTNAME": "Schuessler",
    "SV_MX_MAIL_PRIMARY": "mo.schuessler@example.com",
    "SV_DISPLAYNAME": "Mo Schüssler",
    "MV_MX_PHONE_ADDITIONAL": {
      "__deferred": {
        "uri": "/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/MV_MX_PHONE_ADDITIONAL"
      }
    },
    "ER_MXREF_MX_ROLE": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/ER_MXREF_MX_ROLE(LINK_ID=1738,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "EntryReference"
      },
      "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
      "LINK_ID": 1738,
      "REFERENCED_ID": 41204,
      "REFERENCED_DISPLAY_NAME": "role 4",
      "VALID_FROM": null,
      "VALID_TO": null,
      "REASON": null,
      "STATE": 4,
      "STATE_TEXT": "Pending",
      "DIRECTLY_ASSIGNED": true,
      "INHERITED": false,
      "OPERATION": null,
      "CONTEXT": null,
      "REFERENCED_ENTRY": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1738,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/REFERENCED_ENTRY"
        }
      },
      "CONTEXT_ENTRY": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1738,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/CONTEXT_ENTRY"
        }
      }
    }
  }
}
```
Response Example 3

```json
{
  "d": {
    "d": {
      "VALID_TO": null,
      "REASON": null,
      "STATE": 3,
      "STATE_TEXT": "Failed",
      "DIRECTLY_ASSIGNED": true,
      "INHERITED": false,
      "OPERATION": null,
      "CONTEXT": null,
      "REFERENCED_ENTRY": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1738,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/REFERENCED_ENTRY"
        },
        "CONTEXT_ENTRY": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1738,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/CONTEXT_ENTRY"
          }
        }
      },
      "metadata": {
        "uri": "/idmrestapi/v2/service/ER_MXREF_MX_ROLE(LINK_ID=1739,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "EntryReference",
        "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
        "LINK_ID": 1739,
        "REFERENCED_ID": 41204,
        "REFERENCED_DISPLAY_NAME": "role 4",
        "VALID_FROM": null,
        "VALID_TO": null,
        "REASON": null,
        "STATE": 1,
        "STATE_TEXT": "OK",
        "DIRECTLY_ASSIGNED": true,
        "INHERITED": false,
        "OPERATION": null,
        "CONTEXT": {
          "ID": 32145,
          "DISPLAY_NAME": "Project X",
          "ENTRYTYPE_DISPLAY_NAME": "Project"
        },
        "REFERENCED_ENTRY": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1739,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/REFERENCED_ENTRY"
          }
        },
        "CONTEXT_ENTRY": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=1739,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/CONTEXT_ENTRY"
          }
        }
      }
    }
  }
}
```
Related Information

Get Attribute Values of an Entry [page 50]

7.5 Update an Entry

Updates an entry specified by its MSKEY with any of the attributes defined by the passed TaskGUID.

Prerequisites

- The task is a UI task.
- The logged-in user is authorized to access this task.
- The attributes being written are not read-only.

**MERGE Method via Method Tunneling through POST**

The odata4j library does not support a pure MERGE request. Therefore, the MERGE operation must be requested using method tunneling through POST including the header `X-HTTP-Method` with the value MERGE. If the update is called via the POST HTTP method with MERGE method tunneling, only the attributes that should be updated have to be passed in the request document. If an attribute that is exposed by the specified task is not passed, the update does not change it.

**Update Attributes with Value Help**

For attributes with value help, you must specify the value key as the new value. If you specify the value text, an error occurs.

**Update Multi-value and Entry Reference Attributes**

You can add and update singles values for a multi-value attribute or an entry reference using a merge entry request. Unspecified values or entry references remain unchanged.

You can specify the `OPERATION` property for single changes in an update request. With this operation you can combine different operations in one request without a BATCH request, as the BATCH request is not fully supported by the odata4j library. With the `OPERATION` property you can add, delete, and modify multi-value or entry reference attributes with a single update request.

**Possible Operations for Entry References or Multi-Value Attributes**

<table>
<thead>
<tr>
<th>Table 5: OPERATION Property Values</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Operation Value</strong></td>
</tr>
<tr>
<td>ADD</td>
</tr>
<tr>
<td>Operation Value</td>
</tr>
<tr>
<td>-----------------</td>
</tr>
</tbody>
</table>
| MOD             | Entry Reference: An entry reference identified by the LINK_ID can be modified.  
Multi-Value: This operation is not supported for multi-value attributes. |
| REPLACE         | When adding an entry reference or multi-value with the REPLACE operation, all current entry references or multi-values (of the specific attribute) are removed and replaced with the one specified. If more entry reference or multi-values are added to the same attribute and at least one has the REPLACE operation, all others also must have the REPLACE operation. |
| DEL             | Entry Reference: An entry reference identified by the LINK_ID property can be deleted.  
Multi-Value: A value of the multi-value attribute identified by the VALUE property can be deleted. |
| DEL_ALL         | If this operation was specified, all entry references or multi-values (of the specific attribute) are deleted. When using the DEL_ALL operation, no other entry references or multi-values (of the specific attribute) can be added in the same HTTP request. If you want to do so, use the REPLACE operator instead. |
| `<null>` or OPERATION property not set | Entry References: If no operation was specified the following applies: LINK_ID specified: A modification of the entry reference is executed. If no LINK_ID was specified the REFERENCED_ID property is mandatory; an ADD operation is executed.  
Multi-value: If you do not indicate any operation, the operation ADD (default) is used. |

**Handling of Entry Reference Attributes**

If the attribute is an entry reference, for example, `MXREF_MX_PRIVILEGE`, the specification of only the attribute name results in an error. When creating an assignment (entry reference), you can specify the following properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK_ID</td>
<td>Must be null or do not specify this property.</td>
</tr>
<tr>
<td>REFERENCED_ID</td>
<td>Mandatory</td>
</tr>
</tbody>
</table>
When modifying an assignment (entry reference), you can specify the following properties. To extend the validity of the assignment, also specify the mandatory `VALID_FROM` property. Even if you do not want to change the current value, you have to set this value explicitly.

Table 7: Assignment Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK_ID</td>
<td>Mandatory</td>
</tr>
<tr>
<td>REFERENCED_ID</td>
<td>Optional</td>
</tr>
<tr>
<td>VALID_FROM</td>
<td>Mandatory</td>
</tr>
<tr>
<td>VALID_TO</td>
<td>Mandatory</td>
</tr>
<tr>
<td>OPERATION</td>
<td>Optional or <code>ADD</code> value.</td>
</tr>
</tbody>
</table>

When deleting an assignment (entry reference), you can specify the following properties.

Table 8: Assignment Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>LINK_ID</td>
<td>Mandatory</td>
</tr>
<tr>
<td>REASON</td>
<td>Optional</td>
</tr>
<tr>
<td>OPERATION</td>
<td>Mandatory, value must equal <code>DEL</code>.</td>
</tr>
</tbody>
</table>

**Handling of Multi-value Attributes**

Changing a multi-value attribute is similar to changing an entry reference attribute. You can only change a single value using a collection access as shown in *Request Example 4*. When creating a value for a multi-value attribute, you can specify the following properties.
Delete an Entry

In contrast to the UI tasks that create an entry and can be identified by the *This task creates a new entry* option, there is no equivalent for tasks that delete an entry. These tasks are normal update tasks, which have a dependent task together with a link to a job that will then delete the entry. Therefore the URI format for updating and deleting an entry is the same.

**URI**

```plaintext
{[* [POST with Merge-Method tunneling] *]
URI = './idmrestapi/v2/service/ET_'EntryType'(ID='Mskey','TASK_GUID='TaskGUID');
EntryType = Edm.String;
Mskey = Edm.Int32;
TaskGUID = Edm.Guid;
}
```

**Note**

The IdStoreId is not required here, because the MSKEY is unique across all Identity Stores.

**Request Parameters**

No request parameters. But a document defining an entry with the attributes has to be passed.
Request Example 1

POST /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'D092EA23-9E5D-4FC7-9467-1E1D0BEB3629')/ HTTP/1.1
X-HTTP-METHOD:MERGE

{
    "SV_MX_LASTNAME" : "Benz",
    "SV_MX_FIRSTNAME" : "Marie",
    "SV_MX_MAIL_PRIMARY" : "marie.benz@example.com",
    "SV_DISPLAYNAME" : "Marie Benz",
    "SV_MX_SALUTATION" : "Ms.",
    "ER_MXREF_MX_ROLE" : [ {
        "LINK_ID" : 1737,
        "REASON" : "Extend validity date for existing assignment",
        "VALID_FROM" : "\Date(1360454400000)\",
        "VALID_TO" : "\Date(1372095999000)\",
    }, {
        "REFERENCED_ID" : 41204,
        "REASON" : "New assignment request with validity",
        "VALID_FROM" : "\Date(1360454400000)\",
        "VALID_TO" : "\Date(1362095999000)\",
    }, {
        "REFERENCED_ID" : 41205,
        "REASON" : "New assignment request with context only",
        "CONTEXT" : {
            "ID" : 78
        }
    }, {
        "LINK_ID" : 1738,
        "REASON" : "Delete an assignment via DEL operator",
        "OPERATION" : "DEL"
    } ],
    "MV_MX_PHONE_ADDITIONAL" : [{
        "VALUE" : "+012345689"
    }, {
        "VALUE" : "+987654321"
    }],
    "MV_MX_FAX_ADDITIONAL" : {
        "VALUE" : "+121212121"
    }
}

i Note

To change the VALID_TO value, you must also indicate the VALID_FROM value. If you do not want to change VALID_FROM value, indicate the current VALID_FROM value.

Request Example 2

POST /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'D092EA23-9E5D-4FC7-9467-1E1D0BEB3629')/ HTTP/1.1
X-HTTP-METHOD:MERGE

{
    "ER_MXREF_MX_ROLE" : {
}
"OPERATION" : "DEL_ALL",
}
"MV_MX_PHONE_ADDITIONAL" : [{
   "VALUE" : "+092345689",
   "OPERATION": "REPLACE"
}]
"VALUE" : "+9187654321",
   "OPERATION": "REPLACE"
}]

Response Types

None.

HTTP Response Codes

- 200 – OK
- 400 Bad Request
   A mandatory attribute of the entry is missing or a read-only attribute of the entry is specified.
- 401 – Unauthorized
   The user is not authenticated.
- 403 – Forbidden
   The user is missing authorizations or the XSRF token is missing.
- 404 – Not found
   The specified task or entry was not found. The response body contains information about the not found entity.

Response Example

HTTP/1.1 204 No Content

Related Information

http://www.odata.org/developers/protocols/operations#UpdatingEntries
Add Attribute Values to an Entry [page 54]
Update Attribute Values of an Entry [page 58]
7.6 Get Attribute Values of an Entry

Returns the value(s) for a single attribute of an entry, specified by the entry’s MSKEY, the GUID of the task containing the attribute, and the attribute name.

The specified attribute has to be exposed by the task, passed via the TaskGUID. The task has to be a UI task and the currently logged-in user must be authorized to access this task.

For accessing the “raw” value of a primitive property, `$value` can be used as described in the OData specification. See also Request Example 2. In Identity Management “Raw” value means the pure string representation of the attribute value, as stored in the Identity Center database (for example, the content of the `aValue/aLong` field in `idmv_value_*` view) value of an attribute.

Multi-value and Entry Reference Attributes

To allow the manipulation (add, modify, delete) of single attribute values and entry references, the access to all multi-value and entry reference attributes is done via an access to an Entity Set (collection of entries).

This means, the GET request for an attribute value on these types of attributes returns a collection of either a value or an entry reference. To retrieve a specific value or reference, specify the key of the reference. For multi-value attributes the key is the value itself; for entry references it is the link ID.

### URI

```
(* [GET] *)
```

- **Single-value Attribute**

  ```
  URI = './idmrestapi/v2/service/ET_' EntryType '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName [''/''$value'' ];
  ```

- **Multi-value Attribute**: Attribute is a collection of entries of type “MultiValueString”, “MultiValueDouble”, or “MultiValueDateTime”

  ```
  URI = './idmrestapi/v2/service/MV_' MultiValue '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName ''(' Value ')'' ;
  ```

- Get all values:

  ```
  URI = './idmrestapi/v2/service/MV_' MultiValue '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName;
  ```

- **Entry Reference Attribute**: Attribute is a collection of entries of type “EntryReference”

  ```
  URI = './idmrestapi/v2/service/ER_' EntryReference '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName ''(' LinkId ')'' ;
  ```

- Get one reference:

  ```
  URI = './idmrestapi/v2/service/ER_' EntryReference '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName ''(' LinkId ')'' ;
  ```

- Get all (direct) references:

  ```
  URI = './idmrestapi/v2/service/ER_' EntryReference '(ID=' Mskey ')
  ',TASK_GUID=' TaskGUID ')/' AttributeName;
  ```
Request Parameters

Only the URI for getting all references allows request parameters to specify which assignments should be returned, for example, current, future, failed or only assignments for a certain context (see Request Example 3).

By default only the directly assigned references are returned.

```plaintext
Parameter    = [ '?' Filter ] [ '&' ShowIndirect ]
Filter        = '$filter=' SearchAttributes;
ShowIndirect    = 'showIndirect'; // also shows indirectly assigned references
SearchAttributes    = KeyValuePair (Junction KeyValuePair)*/;
Junction        = 'and';
KeyValuePair    = PropertyName Comperator PropertyValue;
Comperator    = Equal | NotEqual | LowerThan | GraterThan;
Equal        = 'eq';
NotEqual    = 'ne';
LowerThan    = 'lt';
GraterThan    = 'gt';
PropertyValue    = literalExpression;
literalExpression    = stringLiteral / dateTimeLiteral / int16Literal / booleanLiteral / nullLiteral
```

For the literalExpression, see the Abstract Type System section on the OData web page.

Request Example 1

Get all additional phone numbers of the user with ID=7894.

```
GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/MV_MX_PHONE_ADDITIONAL  HTTP/1.1
```

Request Example 2

```
GET /idmrestapi/v2/service/ER_MXREF_MX_ROLE(LINK_ID=1755,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE') HTTP/1.1
```
Request Example 3

GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/BI_MX_USER_PICTURE/$value

Request Example 4

GET /idmrestapi/v2/service/BI_MX_USER_PICTURE(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/$value

Request Example 5

Get direct and indirect assignments of the user with ID=7894.

GET /idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?Assignments&ShowIndirect

Response Types

- application/json
- The type of the attribute when using $value (for example, text/plain for Edm.String).

HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  - If the user is not authenticated.
- 403 - Forbidden
  - If the user is not authorized to access this attribute.
- 404 - Not Found
  - If the requested attribute does not exist in the schema of the entry.

Response Example 1

```json
{
    "d": {
```

<table>
<thead>
<tr>
<th>PUBLIC</th>
<th>Entry Access URIs</th>
</tr>
</thead>
<tbody>
<tr>
<td>52</td>
<td></td>
</tr>
</tbody>
</table>
Response Example 2

```
{ "d": {
  "results": [ {
    "__metadata": {
      "uri": "http://idmrestapi/v2/service/ER_MXREF_MX_ROLE(LINK_ID=4711,TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
      "type": "EntryReference"
    },
    "LINK_ID": 4711,
    "REFERENCED_ID": 40814,
    "REFERENCED_DISPLAY_NAME": "Project Role",
    "CONTEXT": {
      "ID": 32145,
      "DISPLAY_NAME": "ProjectX",
      "ENTRYTYPE_DISPLAY_NAME": "PROJECT"
    },
    "STATE": 4,
    "STATE_TEXT": "Failed",
    "REASON": "User is project lead",
    "VALID_FROM": "\Date(1360454400000)\",
    "VALID_TO": "\Date(1362095999000)\"
  } ]
}
```
Response Example 3

```
{ "d" : { 
  "results" : [ { 
    "__metadata" : { 
      "uri" : "/idmrestapi/v2/service/MV_MX_PHONE_ADDITIONAL(ID=7894, TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE',VALUE='+496123123456789'), 
      "type" : "MultiValueString"
    }, 
    "VALUE" : "+496123123456789" 
    "OPERATION" : null }, 
    { 
      "__metadata" : { 
        "type" : "MultiValueString"
      }, 
      "VALUE" : "+49612319999999" 
      "OPERATION" : null }
  ] }
}
```

Related Information

http://www.odata.org/developers/protocols/json-format#RepresentingTheRawValueOfAProperty

http://www.odata.org/media/16352/%5Bms-odata%5D.pdf

7.7 Add Attribute Values to an Entry

Adds a single value or reference for a single attribute of an entry, specified by the entry’s ID, the GUID of the task containing the attribute, and the attribute name.

The specified attribute must not be read-only and has to be exposed by the task, passed via the TaskGUID. The task must be a UI task and the currently logged-in user must be authorized to access this task.

Note

You can only add values to multi-value attributes or entry reference attributes.
Single Value Entry Reference Attributes

Entry reference attributes not marked as multi-value attributes are represented the same way as multi-value entry reference attributes. However, if the attribute already has one entry reference, adding another entry reference fails.

Adding Multiple Values or References

To add multiple values to a multi-value attribute or multiple references to an entry reference attribute, specify the items in an array list.

Adding Binary Data

Media link attributes are filled in the same way as single value attributes. Insert the binary file content into the body of the POST request. Set the content type of the request accordingly. For example, if you post the content of a GIF image, you have to set the content type of the request to image.gif. To specify a file name, add a header with key Slug and the file name as its value to the request. If you do not pass the file name with the Slug header, a file name is computed based on the content type. For example, the content type image/gif results in a computed file name image/gif.

Note
Do not use an image upload based on HTML form data submit (<form ... enctype="multipart/form-data"> <input ... type="file">...). If you do, it is impossible to pass the mandatory XSRF-token.

By default, the size of the uploaded file is limited to 2097152 bytes (2 MB). The upload of larger files is blocked and results in HTTP response 403. To allow the upload of larger files, adjust the value of the v2.MaximumStorableFileSize application property as necessary.

URI

(* [POST] *)
Multi value attribute:
- Attribute is a collection of entries of type “MultiValueString”, “MultiValueDouble”, or “MultiValueDateTime”
- Add one or more values:
  URI = './idmrestapi/v2/service/ET_'EntryType'(ID='Mskey',TASK_GUID='TaskGUID')/MV_'AttributeName;

Entry reference attribute:
- Attribute is a collection of entries of type “EntryReference”
  - Add one or more references:
Request Parameters

No request parameters. But a document defining the new attribute value or entry reference has to be passed.

Request Example 1

```plaintext
POST/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'D092EA23-9E5D-4FC7-9467-1E1D0B63629')/MV_MX_PHONE_ADDITIONAL HTTP/1.1

[{
   "VALUE":"+4989123456"
},
{
   "VALUE":"+4977646433"
}]
```

Request Example 2

```plaintext
POST/idmrestapi/v2/service/ET_MX_PERSON(ID=7894,TASK_GUID=guid'D092EA23-9E5D-4FC7-9467-1E1D0B63629')/ER_MXREF_MX_ROLE HTTP/1.1

{
   "REFERENCED_ID":40814,
   "CONTEXT":{
      "ID":32145
   },
   "REASON":"User is Project Lead",
   "VALID_FROM":"/Date(1358467200000)="/",
   "VALID_TO":"/Date(1369871999000)="/"
}
```
Response Types

- Content-Type: application/json

HTTP Response Codes

- 201 - Created
  The creation of the attribute value or reference was successful. The created entity is returned in the response.
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  The creation of the value or reference is not allowed for various reasons. Possible reasons are:
  - Missing mandatory parameter.
  - Conflicting unique attribute, that is, the value or reference already exists.
  - Conflicting assignment range.
  - Attribute is a single value entry reference and it already has one reference.
  The error code (if available) and the error message from SAP Identity Management and/or the database are returned in the response body.
- 404 - Not found
  The specified task or entry was not found.

Response Example 1

```
HTTP/1.1 201 Created

{"d":{
  "results":{
    "__metadata":{
      "uri":"/idmrestapi/v2/service/MV_MX_PHONE_ADDITIONAL(ID=7894,TASK_GUID=guid'D092EA23-9E5D-4FC7-9467-1E1D0B8E3629', VALUE='+4989123456')",
      "type":"MultiValueString",
    },
    "VALUE":"+4989123456"
  }
}}
```

Response Example 2

```
HTTP/1.1 201 Created

{"d":{
}}
```
7.8 Update Attribute Values of an Entry

Updates the value for a single attribute of an entry, specified by the entry’s ID, the GUID of the task containing the attribute, and the attribute name.

The specified attribute must not be read-only and has to be exposed by the task, passed via the TaskGUID. The task must be a UI task, and the currently logged-in user must be authorized to access this task.

**Note**

Updating single attribute values (except media data) is not supported in the initial version of the REST interface. This is only possible by updating the entries attributes, see [Update an Entry](#).

Update Attributes with Value Help

For attributes having a value help, you must specify the value key as the value. If you specify the value text, an error occurs.

**URI**

```
(* [POST with MERGE method tunneling] *)
```
URI = './idmrestapi/v2/service/SV_'EntryType'(ID='Mskey',TASK_GUID='TaskGUID')/
AttributeName ['/$value' ];

Multi value attribute:
Attribute is a collection of entries of type "MultiValueString", "MultiValueDouble",
"MultiValueDate"
- Update one value:
URI = './idmrestapi/v2/service/MV_'AttributeName'(ID='Mskey',TASK_GUID='TaskGUID', VALUE='Value')';

Entry reference attribute:
Attribute is a collection of entries of type "EntryReference"
- Update one reference:
URI = './idmrestapi/v2/service/ER_'AttributeName'(ID='Mskey',TASK_GUID='TaskGUID', LINK_ID='LinkId')';

EntryType = Edm.String;
TaskGUID = Edm.Guid;
Mskey = Edm.Int32;
AttributeName = Edm.String;
Value = Edm.String | Edm.Double | Edm.DateTime;
LinkId = Edm.Int32;

Media attribute
URI = './idmrestapi/v2/service/ET_'EntryType'(ID='Mskey',TASK_GUID='TaskGUID')/BI_'AttributeName;
URI = './idmrestapi/v2/service/BI_'AttributeName'(ID='Mskey',TASK_GUID='TaskGUID')

Request Parameters

No request parameters. But a document defining the specified entry attribute has to be passed.

Response Types

None.

HTTP Response Codes

- 204 - No Content
  The update of the entry attribute was successful. No response body is returned.
Response Example

HTTP/1.1 204 No Content

Related Information

Add Attribute Values to an Entry [page 54]
Update an Entry [page 43]

7.9 Search Entry References

There are several ways to search for references of an Identity Management attribute of the Entry reference type.

To search for entry references available at a certain entry, for instance, role assignments for a certain user, you can use custom query options. Using these query options, you can perform the following searches:

- **Status Search**
  You can search by the status of the entry reference. This search type can be combined with either basic search or advanced search.

- **Basic Search**
  You can perform a basic search on the specific attributes (MSKEYVALUE and DISPLAYNAME) of the referenced entry, for instance, the DISPLAYNAME of the referenced role.

- **Advanced Search**
  You can perform an advanced search on the DISPLAYNAME attribute of a referenced entry.

URIs

```
/* [GET] */
- Direct entry reference search:
  URI = './idmrestapi/v2/service/ER_ AttributeName '?ID=' Mskey '&TASK_GUID='
  TaskGUID
- Search entry references for an entry attribute:
  URI = './idmrestapi/v2/service/ET_ EntryTypeName '(ID=' Mskey ',TASK_GUID='
  TaskGUID ')'/ER ' AttributeName
- Get an entry and search for an expanded entry reference attribute:
  URI = './idmrestapi/v2/service/ET_ EntryTypeName '(ID=' Mskey ',TASK_GUID='
  TaskGUID ')?$expand=ER.' AttributeName
  AttributeName = Name of an attribute that has the data type “Entry reference” in the Identity Store schema;
  EntryTypeName = Name of an entry type (MX_ENTRYTYPE) that exists in the Identity Store schema;
  Mskey = Edm.Int32; (* ID of an entry *)
```
Request Parameters

**Status Search**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>= [ StatusFilter ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>StatusFilter</td>
<td>= 'status=' StatusValue;</td>
</tr>
<tr>
<td>StatusValue</td>
<td>= 'ASSIGNED'</td>
</tr>
</tbody>
</table>

- If no status filter and basic filter is specified, all assignments are returned.
- Status "ASSIGNED": All currently active assignments.
- Status "NOT_ASSIGNED":
  - Pending: Assignments are currently (de)provisioned or not yet approved. This explicitly does not include future assignments.
  - The assignment failed.
  - The assignment was rejected.
- Status "FUTURE": Pending assignments, because the valid-from date is still in the future.
- Status "ASSIGNED AND PENDING":
  - Assigned OK
  - Pending: Waiting for approval
  - Pending: Assignments are currently provisioned.
  - Future
  - Fails: waiting for another provisioning attempt
  - Rejected is excluded

**Basic Search**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>= [ BasicFilter ] [ StatusFilter ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>BasicFilter</td>
<td>= 'filterBasic=' SearchValue;</td>
</tr>
<tr>
<td>SearchValue</td>
<td>= Complete or part (with wildcard character star [*]) of the DISPLAYNAME or MSKEYVALUE</td>
</tr>
<tr>
<td>StatusFilter</td>
<td>= 'status=' StatusValue;</td>
</tr>
<tr>
<td>StatusValue</td>
<td>= 'ASSIGNED'</td>
</tr>
</tbody>
</table>

*Note*

The wildcard character star [*] is allowed in the value of "filterBasic".
### Advanced Search

Parameter search can be configured as:

- **StatusFilter**
  - `status='StatusValue';`
  - Valid values: `ASSIGNED` | `NOT_ASSIGNED` | `FUTURE` | `ASSIGNED_AND_PENDING`

- **Filter**
  - `$filter='DisplayNameFilter ComparatorSearchValue';`
  - `Comparator` = `eq`;
  - `SearchValue` = Complete or part (with wildcard character star `[*]`) of the DISPLAYNAME in single quotes

**Note**
- Currently, only the search for the referenced display name is supported.
- The wildcard character star `[*]` is allowed in the search value of `$filter`.

### Request Example 1


### Request Example 2

GET `/idmrestapi/v2/service/ET_MX_PERSON(ID=25,TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FM*')/ER_MXREF_MX_PRIVILEGE?status=FUTURE`

### Request Example 3

GET `/idmrestapi/v2/service/ET_MX_PERSON(ID=25,TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FM*)?$expand=ER_MXREF_MX_PRIVILEGE&status=ASSIGNED`

### Request Example 4

GET `/idmrestapi/v2/service/ET_MX_PERSON(ID=25,TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FFFF-FM*)?$expand=ER_MXREF_MX_PRIVILEGE&status=ASSIGNED&filterBasic=M*`
Request Example 5

GET/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE?ID=25&TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF'?$filter=REFERENCED_DISPLAY_NAMEeq'M*'

Response Types

Content-Type: application/json | application/atom+xml

HTTP Status Codes:

- **200 OK**
  - Response Body: Collection of entities with the type EntryReference.

- **400 Bad Request**
  - The “status” parameter does not have one of the predefined values.
  - The given EntryTypeName does not fit to the given MSKEY.

- **403 Forbidden**
  - Logged-in user is not allowed to search for entry references.

- **404 Not Found**
  - AttributeName, EntryTypeName, Mskey, or TaskGUID does not exist.

Response Example 1

```
{  
  "d": {  
    "results": [  
      {  
        "__metadata": {  
          "uri": "http://10.66.178.212:53000/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=5478,TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF')",  
          "type": "IdMRestModel.EntryReference"  
        },  
        "TASK_GUID": "FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF",  
        "LINK_ID": 5478,  
        "REFERENCED_ID": 8,  
        "REFERENCED_DISPLAY_NAME": "MX_PRIV:TRANSPORT:IMPORT",  
        "VALID_FROM": "\Date(1359590400000)\",  
        "VALID_TO": null,  
        "REASON": null,  
        "STATE": 513,  
        "STATE_TEXT": "Pending",  
        "CONTEXT": null  
      },  
      {  
        "__metadata": {  
          "uri": "http://10.66.178.212:53000/idmrestapi/v2/service/ER_MXREF_MX_PRIVILEGE(LINK_ID=5479,TASK_GUID=guid'FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF")",  
          "type": "IdMRestModel.EntryReference"  
        },  
        "TASK_GUID": "FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF",  
        "LINK_ID": 5479,  
        "REFERENCED_ID": 8,  
        "REFERENCED_DISPLAY_NAME": "MX_PRIV:TRANSPORT:IMPORT",  
        "VALID_FROM": "\Date(1359590400000)\",  
        "VALID_TO": null,  
        "REASON": null,  
        "STATE": 513,  
        "STATE_TEXT": "Pending",  
        "CONTEXT": null  
      }  
    ]  
  }  
}
```
The data displayed for Response Example 1 are also valid as part of the responses for the two other request examples.

7.10  Context Usage

The assignment of privileges and roles supports the specification of a context. The same role or privilege can be assigned with different contexts. Additionally, it can also be assigned without a context. If an assignment is done without a context, the value for the context must be `null`.

7.11  Virus Scanning

The Virus Scan Interface of the AS Java can scan binary data for viruses during the upload. The following prerequisites must be met:

- A third-party virus scanner is installed. As it is not part of the SAP NetWeaver AS for Java it has to be licensed separately.
- A default profile must be configured in the Virus Scan Interface (VSI) infrastructure.

During the deployment of the REST application, two VSI profiles are created, unless these profiles have already been created during a former deployment:

- The `idmRest_ImageUpload` profile used for binary attributes with the Picture presentation type.
  The profile contains a default configuration for virus scanning, MIME type matching (tests if the file content matches the file name extension) and content type filtering based on a MIME type whitelist.
- The `idmRest_FileUpload` used for any other binary attributes.
  The profile is a default configuration for virus scanning, MIME type matching and active content blocking. This prevents, for example, uploading XMLs that contain Javascript.

The profiles must be activated manually in the SAP NetWeaver Administrator, otherwise no checks are performed. If the VSI scan detects any security issue in the uploaded data, the request results in a `Forbidden` response with a short description of the reason.

For more information, see the Virus Scan Interface documentation in the SAP Library.
Related Information

Java-specific Configuration of the Virus Scan Interface
8 Task Access URIs

The URIs described in the sections below allow reading Identity Management Tasks that are defined as user interface tasks (UI). You can search for tasks, read details for a certain task or read task attributes. The query only returns those tasks to which the user has access.

8.1 Get All Self-Service Tasks

Returns a collection of all available self-service Tasks.

Self-service tasks are user interface tasks that users can execute for their own entry, for example, to change their personal data. These tasks operate on the entry type of the logged-in user. All logged-in users who are assigned the rest_V2 UME action have access to this task and can execute it for their own entry only, typically MX_PERSON.

URI

(* [GET] *)

URI = './idmrestapi/v2/service/SELFSERVICETASKS';

Request Parameters

None

Request Example

GET /idmrestapi/v2/service/SELFSERVICETASKS HTTP/1.1

Response Types

Content-Type: application/json
HTTP Response Codes

- 200 OK
  Response Body: Collection(Task)

Response Example

```json
{
  "d": {
    "results": [{
      "__metadata": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')",
        "type": "Task"
      },
      "TASK_ID": 1318,
      "TASK_GUID": "494E68D9-ECCE-41F5-AE3D-B93D23F54E70",
      "TASK_DISPLAY_NAME": "Change Own Data",
      "TASK_GROUP": "Identity",
      "TASK_HEADER": null,
      "TASK_FOOTER": "Task to change your data",
      "TASK_ATTRIBUTES": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/TASK_ATTRIBUTES"
        }
      },
      "CURRENT_ENTRY": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/CURRENT_ENTRY"
        }
      }
    }, {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')",
        "type": "Task"
      },
      "TASK_ID": 30,
      "TASK_GUID": "494E68D9-ECCE-41F5-AE3D-B93D23F54E70",
      "TASK_DISPLAYNAME": "Request Role Assignment",
      "TASK_GROUP": "Identity",
      "TASK_HEADER": null,
      "TASK_FOOTER": "Task for requesting new roles",
      "TASK_ATTRIBUTES": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/TASK_ATTRIBUTES"
        }
      },
      "CURRENT_ENTRY": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/CURRENT_ENTRY"
        }
      }
    }
  }
}
```
8.2 Get a Specific Self-Service Task

Returns only one specific self-service task indicated by its task GUID.

If the provided task GUID is not from a self-service task, the call returns an error response.

Because of the special semantics of these tasks, the entry access is limited to the currently logged-in user. Therefore, the returned task entity contains the additional dynamic property “CURRENT_ENTRY” with the entry access URI for the current user and this task, as specified in Get Specific Entry under Response Examples.

URI

```plaintext
(* [GET] *)
URI      = './idmrestapi/v2/service/SELFSERVICETASKS('TaskGUID')';
TaskGUID = Edm.Guid;
```

Request Parameters

None

Request Example 1

GET /idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70') HTTP/1.1

Request Example 2

Get specified self-service task and execute it (get attributes + values) for the current user.

GET /idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')?$expand=CURRENT_ENTRY HTTP/1.1

Response Types

- Content-Type: application/json
HTTP Response Codes

- **200 OK**
  Response Body: Task

- **403 Forbidden**
 ○ Task GUID is not from a self-service task.

**Response Example 1**

```json
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')",
        "type": "Task"
      }
    },
    "TASK_ID": 1318,
    "TASK_GUID": "GUID-GUID-123",
    "DISPLAYNAME": "Change Own Data",
    "TASK_GROUP": "Identity",
    "TASK_HEADER": null,
    "TASK_FOOTER": "Task to change your data",
    "TASK_ATTRIBUTES": {
      "__deferred": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/TASK_ATTRIBUTES"
      }
    },
    "CURRENT_ENTRY": {
      "__deferred": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')/CURRENT_ENTRY"
      }
    }
  }
}
```

**Response Example 2**

```json
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/SELFSERVICETASKS(guid'494E68D9-ECCE-41F5-AE3D-B93D23F54E70')",
        "type": "Task"
      }
    },
    "TASK_ID": 1318,
    "TASK_GUID": "GUID-GUID-123",
    "DISPLAYNAME": "Change Own Data",
    "TASK_GROUP": "Identity",
    "TASK_HEADER": null,
    "TASK_FOOTER": "Task to change your data",
    "TASK_ATTRIBUTES": {
```
```
Related Information

Get Specific Entry [page 38]

8.3 Get All Task Attributes

Return information about all the attributes that a task specified by its TaskGUID exposes of an entry type. For details about the properties of a task attribute, see Get a Specific Task Attribute.

URI

(* [GET]*)

URI = './idmrestapi/v2/service/TASK_ATTRIBUTES';

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Filter</th>
</tr>
</thead>
<tbody>
<tr>
<td>TaskGUIDFilter</td>
<td>'TASK_GUID=' TaskGUID</td>
</tr>
<tr>
<td>TaskGUID</td>
<td>Edm.Guid</td>
</tr>
</tbody>
</table>
Note

TASK_GUID is a mandatory filter attribute.

Request Example

GET /idmrestapi/v2/service/TASK_ATTRIBUTES?TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE'
HTTP/1.1

Response Types

Content-Type: application/json | application/atom+xml

HTTP Response Codes

- 200 OK
  Response Body: Collection(TaskAttribute)
- 403 Forbidden
  Logged-in user is not allowed to access the task.
- 404 Not Found
  The specified task does not exist.

Response Example

```json
{
  "d": {
    "results": [
      {
        "__metadata": {
          "uri": "!/idmrestapi/v2/service/TASK_ATTRIBUTES(ATTRIBUTE_NAME='MSKEYVALUE', TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
          "type": "IdMRestModel.TaskAttribute"
        },
        "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
        "ATTRIBUTE_NAME": "MSKEYVALUE",
        "DISPLAY_NAME": "Unique ID",
        "DEFAULT_VALUE": null,
        "HIDE_INPUT": false,
        "TOOLTIP": "",
        "VALUEHELP": false,
        "READONLY": true,
        "MANDATORY": false,
        "ALLOWED_VALUES": [
```
Related Information

Get a Specific Task Attribute [page 73]
8.4 Get a Specific Task Attribute

Returns only a specific task attribute specified by the TaskGUID and the task attribute name.

The properties of a task attribute are a subset of the properties of an attribute exposed by the schema information requests. Only the properties are contained that can be overwritten on task level. For the complete list, see the metadata.xml file. If a property is not set on the task level, the property value of the attribute in the schema is returned as a fallback.

URI

```plaintext
(* [GET] *)
URI                  = './idmrestapi/v2/service/
TASK_ATTRIBUTES(ATTRIBUTE_NAME='TaskAttributeName',TASK_GUID=' TaskGUID ')';
TaskGUID             = Edm.Guid;
TaskAttributeName    = Edm.String;
```

Request Parameters

Supported Query Options:

- $expand: To inline deferred allowed values.

Request Example 1

```
GET /idmrestapi/v2/service/
TASK_ATTRIBUTES(ATTRIBUTE_NAME='MX_FIRSTNAME',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')   HTTP/1.1
```

Request Example 2

```
GET /idmrestapi/v2/service/
TASK_ATTRIBUTES(ATTRIBUTE_NAME='MX_SALUTATION',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')?$expand=ALLOWED_VALUES   HTTP/1.1
```
Response Types

Content-Type: application/json | application/atom+xml

HTTP Response Codes

- 200 OK
  Response body: TaskAttribute
- 403 Forbidden
  Logged-in user is not allowed to access the task.
- 404 Not Found
  The specified task does not exist.

Response Example 1

```json
{
  "d": {
    "results": {
      "__metadata": {
        "uri": "/idmrestapi/v2/service/TASK_ATTRIBUTES(ATTRIBUTE_NAME='MX_FIRSTNAME',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "IdMRestModel.TaskAttribute"
      },
      "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
      "ATTRIBUTE_NAME": "MX_FIRSTNAME",
      "DISPLAY_NAME": "First Name",
      "DEFAULT_VALUE": null,
      "HIDE_INPUT": false,
      "TOOLTIP": "",
      "VALUEHELP": false,
      "READONLY": true,
      "MANDATORY": false,
      "ALLOWED_VALUES": {
        "__deferred": {
          "uri": "/idmrestapi/v2/service/TASK_ATTRIBUTES(ATTRIBUTE_NAME='MX_FIRSTNAME',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')/ALLOWED_VALUES"
        }
      }
    }
  }
}```
Response Example 2 ($expand=ALLOWED_VALUES)

```json
{
  "d": {
    "results":{
      "__metadata":{
        "uri": "/idmrestapi/v2/service/TASK_ATTRIBUTES(ATTRIBUTE_NAME='MX_SALUTATION',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE')",
        "type": "IdMRestModel.TaskAttribute"
      },
      "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
      "ATTRIBUTE_NAME": "MX_SALUTATION",
      "DISPLAY_NAME": "Title",
      "DEFAULT_VALUE": null,
      "HIDE_INPUT": false,
      "TOOLTIP": "",
      "VALUEHELP": true,
      "READONLY": true,
      "MANDATORY": false,
      "ALLOWED_VALUES": {
        "results": {
          "__metadata":{
            "uri": "/idmrestapi/v2/service/ALLOWED_VALUES(ATTRIBUTE_NAME='MX_SALUTATION',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE',VALUE_KEY='0001')",
            "type": "IdMRestModel.AllowedValue"
          },
          "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
          "ATTRIBUTE_NAME": "MX_SALUTATION",
          "VALUE_KEY": "0001",
          "VALUE_TEXT": "Ms."
        },
        {
          "__metadata":{
            "uri": "/idmrestapi/v2/service/ALLOWED_VALUES(ATTRIBUTE_NAME='MX_SALUTATION',TASK_GUID=guid'35B264BF-A75A-447D-A4EA-7894725245CE',VALUE_KEY='0002')",
            "type": "IdMRestModel.AllowedValue"
          },
          "TASK_GUID": "35B264BF-A75A-447D-A4EA-7894725245CE",
          "ATTRIBUTE_NAME": "MX_SALUTATION",
          "VALUE_KEY": "0002",
          "VALUE_TEXT": "Mr."
        }
      }
    }
  }
}
```
8.5 Get Allowed Values of a Specific Task Attribute

This access gets a list of the permissible values for a certain task attribute.

The property “ALLOWED_VALUES” specifies the values that are allowed for a specific attribute. It returns the evaluated list of all allowed values dependent on the type of the allowed (legal) values:

- Values: The specified list of values is returned.
- SQL Query: The result list of the query is returned.
- Value help: Returns the list of all keys with their descriptions from the value help table for this attribute.
- Task: The result of the task is returned.

If no allowed values are defined for the attribute on the UI task level, the allowed values defined on the attribute in the identity store schema are returned.

**Note**

If the attribute is of data type entry reference, the number of returned allowed values is affected by the Max search result size setting. You can find more information in Search Entries [page 33].

The value help can be language-dependent, for example, salutation, title, and job title. The values returned are returned in the language of the currently logged-in user. For more information about how the language is determined, see also Localization.

**URI**

```
(* [GET] *)
URI       = './idmrestapi/v2/service/ALLOWED_VALUES
```

**Request Parameters**

```
Parameter        = '?' Filter
Filter           = '$filter=' MandatoryFilter [Junction OptionalFilter];
MandatoryFilter  = TaskGUIDFilter Junction TaskAttributeFilter;
OptionalFilter   = KeyValuePair [Junction KeyValuePair];
TaskGUIDFilter   = 'TASK_GUID' Comparator TaskGUID
TaskAttributeFilter = 'ATTRIBUTE_NAME' Comparator AttributeName
KeyValuePair    = FilterPropertyName Comparator SearchValue;
FilterPropertyName = 'VALUE_KEY' | 'VALUE_TEXT'
UserMskeyFilter  = 'USERMSKEY' Comparator UserMskey
Junction         = 'and';
Comparator       = 'eq';
TaskGUID         = Edm.Guid;
```
AttributeName     = "" Edm.String "";
SearchValue        = "" Edm.String ""; (* * is a wildcard *)
UserMskey          = "" Edm.String "";

**Note**
- The filter parameter is case sensitive, except for the search value.
- TASK_GUID and ATTRIBUTE_NAME are mandatory filter attributes.
- The URI has to be URL escaped to replace special characters, like the space with "%20".

**Request Example**

```plaintext
[GET]
/idmrestapi/v2/service/ALLOWED_VALUES?$filter=ATTRIBUTE_NAME eq 'MX_SALUTATION' and TASK_GUID eq guid'35B264BF-A75A-447D-A4EA-7894725245CE'
```

**Response Example**

```json
}
```
9 Approval Access URIs

You can use the following URIs to access SAP Identity Management approval items, for example, changes to entries that do not take effect immediately, but need the approval of an authorized person. You can query for approvals that the currently logged-in user can process, by accessing these items and approving or declining them. Some of the general URIs can also allow you to access the attestation items.

Related Information

Attestation Access URIs [page 85]

9.1 Get Open Approval Requests for Current User

This GET request returns a list containing approval items assigned to the logged-in user. The list includes all of the information about the approval item that is needed for an approver to decide if the item can be approved. The list can also contain information about the attestation task.

URI

(* [GET] *)
URI = './idmrestapi/v2/service/TaskCollection'

Request Parameters

None.

Request Example

[GET]/idmrestapi/v2/service/TaskCollection
Response Types

Content-Type: application/json

HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  If the user is not authorized to query for open approvals.
- 405 - Method Not Allowed
  If the URI was not called with a GET request.

Response Example

```json
{
  "d":{
    "results":[
      {
        "metadata" :{
          "uri" : "/idmrestapi/v2/service/TaskCollection(InstanceID='1x7',SAP__Origin='IDM')",
          "type" : "TASKPROCESSING.Task"
        },
        "SAP__Origin" : "IDM",
        "InstanceID" : "1x7",
        "TaskDefinitionID" : "1",
        "CreatedBy" : "Maximilian Muster",
        "CreatedOn" : "/Date(1362355200000)/",
        "ExpiryDate" : "/Date(1364947200000)/",
        "CREATED_BY_ID" : 318,
        "REQUESTED_ENTRY_ID" : 218,
        "REQUESTED_ENTRY_NAME" : "SAP_SDIC_SUPER_ADMINISTRATOR-Role (ASJAVA730)",
        "VALID_FROM" : "/Date(1362355200000)/",
        "VALID_TO" : "/Date(1364256000000)/",
        "REASON" : "Need SDIC Super Admin rights",
        "CONTEXT" : null,
        "CustomAttributeData" :{
          "_deferred" :{
            "uri" : "/idmrestapi/v2/service/TaskCollection(InstanceID='1x7',SAP__Origin='IDM')/CustomAttributeData"
          }
        },
        "TaskDefinitionData" :{
          "_deferred" :{
            "uri" : "/idmrestapi/v2/service/TaskCollection(InstanceID='1x7',SAP__Origin='IDM')/TaskDefinitionData"
          }
        },
        "REFERENCED_ENTRY" :{
          "_deferred" :{
```
"uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x7',SAP__Origin='IDM')/REFERENCED_ENTRY"},
"CONTEXT_ENTRY":{
  "__deferred":{
    "uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x7',SAP__Origin='IDM')/CONTEXT_ENTRY"
  }
},
"__metadata":{
  "uri": "/idmrestapi/v2/service/TaskCollection(InstanceID='1x3',SAP__Origin='IDM')",
  "type":"TASKPROCESSING.Task"
},
"SAP__Origin":"IDM",
"InstanceID":"1x3",
"TaskDefinitionID":"1",
"CreatedBy":"Hans Müller",
"CreatedOn": "/Date(1362009600000)="/",
"ExpiryDate": "/Date(1364601600000)="/",
"CREATED_BY_ID":318,
"REQUESTED_ENTRY_ID":195,
"REQUESTED_ENTRY_NAME":"Administrators-Group (ASJAVA730)",
"VALID_FROM":null,
"VALID_TO": "/Date(1363824000000)="/",
"REASON":null,
"CONTEXT": {
  "ID": 2,
  "DISPLAY_NAME":"ProjectX",
  "ENTRYTYPE_DISPLAY_NAME": "Project "
},
"CustomAttributeData":{
  "__deferred":{
    "uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x3',SAP__Origin='IDM')/CustomAttributeData"
  }
},
"TaskDefinitionData":{
  "__deferred":{
    "uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x3',SAP__Origin='IDM')/TaskDefinitionData"
  }
},
"REFERENCED_ENTRY":{
  "__deferred":{
    "uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x3',SAP__Origin='IDM')/REFERENCED_ENTRY"
  }
},
"CONTEXT_ENTRY":{
  "__deferred":{
    "uri":"/idmrestapi/v2/service/TaskCollection(InstanceID='1x3',SAP__Origin='IDM')/CONTEXT_ENTRY"
  }
}
"__count":"2"}
9.2 Process an Approval

An open approval can be either approved or declined by the currently logged-in user, if that user is the currently assigned processor for the approval. Approval operations are executed using the decision service operation. The available decision options for a specific approval request can be retrieved by executing the DecisionOptions service operation.

9.2.1 Get Valid Decision Options for Approval

Valid decision options can be retrieved by executing the DecisionOptions service operation. You must specify the request ID (InstanceID) as a parameter. The DecisionKey property with the respective value must be specified as a parameter for the decision service operation. The \textit{1x} part of the instance ID shows that the request is for an approval task.

**URI**

\[
(* \ [GET] \ *)
\]

\texttt{URI} = './idmrestapi/v2/service/DecisionOptions'

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>'=' RequestId</td>
</tr>
<tr>
<td>RequestId</td>
<td>= Edm.String;</td>
</tr>
</tbody>
</table>

**Request Example**

\[
[GET] /idmrestapi/v2/service/DecisionOptions?InstanceID='1x4'
\]
Response Types

Content-Type: application/json

HTTP Response Codes

- **200 - OK**
- **401 - Unauthorized**
  If the user is not authenticated.
- **405 - Method Not Allowed**
  If the URI was not called with a GET request.

Response Example

```
{
  "d": [
    {
      "SAP__Origin": "IDM",
      "InstanceID": "1x4",
      "DecisionKey": "APPROVE",
      "DecisionText": "Approve",
      "CommentMandatory": false,
      "Nature": "POSITIVE"
    },
    {
      "SAP__Origin": "IDM",
      "InstanceID": "1x4",
      "DecisionKey": "DECLINE",
      "DecisionText": "Decline",
      "CommentMandatory": false,
      "Nature": "NEGATIVE"
    }
  ]
}
```

9.2.2 Execute Approval Decision

*Get Valid Decision Options* describes how to retrieve a list of available decision options. After choosing a decision option, you must specify the `DecisionKey` and the request ID (`InstanceID`) of the approval request as a parameter of the decision service operation. The `1x` part of the `InstanceID` indicates that this is an approval request. You can also specify a comment. The `DecisionOptions` service operation indicates whether the `Comments` parameter is mandatory. If the approval decision succeeds, the approval request (task) is returned to the client.
URI

```text
(* [POST] *)
URI = './idmrestapi/v2/service/Decision'
```

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>'InstanceID=' RequestId</td>
</tr>
<tr>
<td>SAP__Origin</td>
<td>'SAP__Origin=' 'IDM'</td>
</tr>
<tr>
<td>DecisionKey</td>
<td>'DecisionKey=' DecisionKeyValue</td>
</tr>
<tr>
<td>Comments</td>
<td>'Comments=' CommentsValue</td>
</tr>
<tr>
<td>RequestId</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DecisionKeyValue</td>
<td>Edm.String</td>
</tr>
<tr>
<td>CommentsValue</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

**Note**

The parameters in the brackets ([ ]) are optional.

Request Example 1

```text
[POST]idmrestapi/v2/service/Decision?InstanceID='1x4'&DecisionKey='APPROVE'
```

Request Example 2

```text
[POST]idmrestapi/v2/service/Decision?
InstanceID='1x4'&DecisionKey='DECLINE'&Comments='Not allowed'
```

Response Types

```text
Content-Type: application/json
```
HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  If the user is not authorized to process these open approvals.
- 404 - Not Found
  If the requested approval item does not exist.
- 405 - Method Not Allowed
  If the URI was not called with a POST request.

Response Example

HTTP/1.1 200 OK
{
  "d":{
    "__metadata": {
      "uri": "/idmrestapi/v2/service/
TaskCollection(InstanceId='1x4',SAP__Origin='IDM'),
  "type": "IdMRestModel.Task"
    },
  "SAP__Origin": "IDM",
  "InstanceId": "1x4",
  "CREATED_BY_ID" : 89,
  "CreatedBy": "Hans Müller",
  "CreatedOn": "/Date(1360800000000)/",
  "REQUESTED_ENTRY_ID": 2441,
  "REQUESTED_ENTRY_NAME": "SAP_ALL-Profile (BCE324)",
  "ExpiryDate": "/Date(1363392000000)/",
  "VALID_FROM": "/Date(1363651200000)/",
  "VALID_TO": "/Date(1364601600000)/",
  "REASON": "Need SDIC Super Admin rights",
  "CONTEXT": {
    "ID": 325,
    "DISPLAY_NAME": "ProjectX",
    "ENTRYTYPE_DISPLAY_NAME": "Project Context"
  },
  "REFERENCED_ENTRY": {
    "__deferred": {
      "uri": "/idmrestapi/v2/service/
TaskCollection(InstanceId='1x4',SAP__Origin='IDM')/REFERENCED_ENTRY"
    }
  },
  "CONTEXT_ENTRY": {
    "__deferred": {
      "uri": "/idmrestapi/v2/service/
TaskCollection(InstanceId='1x4',SAP__Origin='IDM')/CONTEXT_ENTRY"
    }
  }
}
10 Attestation Access URIs

You can use the following URIs to access attestation items of SAP Identity Management, for example, user assignments to specific roles that need to be confirmed periodically by an authorized person. The attester can confirm or reject the user assignment. He or she can also delegate this responsibility to another attester. The attestation tasks are part of the same resource as the approval tasks.

Related Information

Approval Access URIs [page 78]

10.1 Get Attestation Requests for Current User

The **GET** request returns a list containing attestation items assigned to the logged-in user. The list includes all of the information about the attested item that is needed for an attester to decide whether to confirm, reject, or delegate the item. The list can also contain information about the approval task.

**URI**

```
(* [GET] *)
URI = './idmrestapi/v2/service/TaskCollection'
```

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>'InstanceID=' RequestId</td>
</tr>
<tr>
<td>SAP_Origin</td>
<td>'SAP_Origin=' 'IDM'</td>
</tr>
<tr>
<td>expand</td>
<td>'expand=' expand_value</td>
</tr>
<tr>
<td>expand_value</td>
<td>Assignments</td>
</tr>
</tbody>
</table>

**Note**

The parameters in the brackets ([ ]) are optional.
Request Example

[GET]/idmrestapi/v2/service/TaskCollection

Request Example for Getting One Attestation Task

The currently logged-in user can see one specific attestation task identified by the InstanceID parameter. The task contains all the attestation resources the user needs for the attestation. The 2x part of the instance ID indicates that this is an attestation task.

- Request Example

  [GET]/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')

Request Example for Getting Assignments

The currently logged–in user can see detailed information about the assignments for all the resources of one specific attestation task identified by InstanceID.

[GET]/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')/?$expand=Assignments

Request Example for Getting Data for All Custom Attributes

The currently logged–in user can see detailed information about the custom attributes for all the resources of one specific attestation task identified by InstanceID.

The data contains detailed information about the following attributes:

- ROLE_NAME
- ROLE_DESCRIPTION
- LAST_ATTESTED_ON
- ASSIGNMENTS_COUNT
- STARTED_BY_ID
- STARTED_BY
- ATTESTATION_TYPE

[GET]/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')/?$expand=CustomAttributeData
Response Types

Content-Type: application/json

HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  If the user is not authorized to query for attestation.
- 405 - Method Not Allowed
  If the URI was not called with a GET request.

Note

The collection UIExecutionLink is not supported as part of the OData model. If you include it in the request, in the HTTP response you will get a Not Found error.

Response Example

```json
{
  "d": {
    "results": [
      {
        "__metadata": {
          "uri": "/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')",
          "type": "TASKPROCESSING.Task"
        },
        "SAP__Origin": "IDM",
        "TaskDefinitionID": "2",
        "InstanceID": "2x11",
        "CreatedOn": "/Date(1382572800000)/",
        "CreatedBy": "<SYSTEM>",
        "CompletionDeadline": "/Date(1382659200000)/",
        "ExpiryDate": "/Date(1382659200000)/",
        "TaskTitle": "Attestation for Role/Privilege Administrator",
        "TaskDefinitionData": {
          "__deferred": {
            "uri": "/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')/TaskDefinitionData"
          }
        },
        "CustomAttributeData": {
          "results": [
            {
              "__metadata": {
                "uri": "/idmrestapi/v2/service/CustomAttribute(InstanceID='2x11',Name='ROLE_NAME',SAP__Origin='IDM')"
              }
            }
          ]
        }
      }
    ]
  }
}
```
Related Information

Approval Access URIs [page 78]

10.2 Process an Attestation

An attestation can be confirmed, rejected, or delegated by the currently logged-in user, if that user is the currently assigned processor for the attestation. Attestation operations are executed using the decision service operation. The available decision options for a specific attestation request can be retrieved by executing the DecisionOptions service operation.

10.2.1 Get Valid Decision Options for Attestation

Valid decision options for attestation can be retrieved by executing the DecisionOptions service operation. You must specify the request ID (InstanceID) as a parameter. The DecisionKey property with the respective value
must be specified as a parameter for the decision service operation. The 2x part of the instanceID shows that the request is for an attestation task.

**URI**

```plaintext
(* [GET] *)
URI = './idmrestapi/v2/service/DecisionOptions'
```

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>'InstanceID=' RequestId</td>
</tr>
<tr>
<td>RequestId</td>
<td>Edm.String;</td>
</tr>
</tbody>
</table>

**Request Example**

```
[GET] /idmrestapi/v2/service/DecisionOptions?InstanceID='2x11'
```

**Response Types**

```
Content-Type: application/json
```

**HTTP Response Codes**

- **200 - OK**
- **401 - Unauthorized**
  - If the user is not authenticated.
- **405 - Method Not Allowed**
  - If the URI was not called with a GET request.

**Response Example**

```
{}
```
10.2.2 Execute Attestation Decision

After you have selected the appropriate `DecisionOptions`, ATTEST or DELEGATE, you can go on to perform the attestation on a specific task, which can be uniquely identified by its instance ID. The value of the `instanceID` parameter starts with 2x, which indicates that this is an attestation task.

**URI**

```
(* [POST] *)
URI = './idmrestapi/v2/service/Decision'
```

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>'InstanceID=' RequestId</td>
</tr>
<tr>
<td>SAP_Origin</td>
<td>'SAP_Origin=' 'IDM'</td>
</tr>
<tr>
<td>DecisionKey</td>
<td>'DecisionKey=' DecisionKeyValue</td>
</tr>
<tr>
<td>Action</td>
<td>'Action=' ActionValue</td>
</tr>
<tr>
<td>LinkId</td>
<td>'LinkId=' LinkIdValue</td>
</tr>
<tr>
<td>DelegateId</td>
<td>'DelegateId=' DelegateIdValue</td>
</tr>
<tr>
<td>Comments</td>
<td>'Comments=' CommentsValue</td>
</tr>
<tr>
<td>RequestId</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DecisionKeyValue</td>
<td>Edm.String</td>
</tr>
<tr>
<td>ActionValue</td>
<td>Edm.String</td>
</tr>
<tr>
<td>LinkIdValue</td>
<td>Edm.String</td>
</tr>
<tr>
<td>DelegateIdValue</td>
<td>Edm.String</td>
</tr>
<tr>
<td>CommentsValue</td>
<td>Edm.String</td>
</tr>
</tbody>
</table>

SAP Identity Management REST Interface Version 2

Attestation Access URIs

PUBLIC 91
Request Example for Positive Attestation

The attester confirms that the attested role belongs to the attested user with the following settings:

- DecisionKey: ATTEST
- Action: CERTIFY
- LinkId: ID of one particular resource to be attested

```
[POST] idmrestapi/v2/service/Decision?InstanceID='2x11'&SAP__Origin='IDM'&DecisionKey='ATTEST'&Action='CERTIFY'&LinkId='31'
```

Request Example for Negative Attestation

The attester rejects that the attested role belongs to the attested user with the following settings:

- DecisionKey: ATTEST
- Action: REJECT
- LinkId: ID of one particular resource to be attested

```
[POST] idmrestapi/v2/service/Decision?InstanceID='2x11'&SAP__Origin='IDM'&DecisionKey='ATTEST'&Action='REJECT'&LinkId='31'
```

Request Example for Delegating Attestation

The attester is not sure if the attested role belongs to the attested user, so he delegates the attestation of one of the assignments to another attester with the following settings:

- DecisionKey: DELEGATE
- DelegateId: ID of the user to whom the attestation is to be delegated
- LinkId: ID of one particular resource to be attested

```
[POST] idmrestapi/v2/service/Decision?InstanceID='2x11'&SAP__Origin='IDM'&DecisionKey='DELEGATE'&DelegateId='30'&LinkId='900'
```

i Note
The LinkId parameter in this example specifies which assignment to be delegated for attestation.
Response Types

Content-Type: application/json

HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  If the user is not authorized to process this task.
- 404 - Not Found
  If the requested attestation item does not exist.
- 405 - Method Not Allowed
  If the URI was not called with a POST request.

Response Example

```json
{
  "d": {
    "__metadata": {
      "uri": "/idmrestapi/v2/service/TaskCollection(InstanceID='2x11',SAP__Origin='IDM')",
      "type": "TASKPROCESSING.Task"
    },
    "SAP__Origin": "IDM",
    "TaskDefinitionID": "2",
    "InstanceID": "2x11",
    "TaskTitle": "Attestation"
  }
}
```

10.3 Get a History of Attestations

You can use this request to display a history log of previous attestations. The log can give you information about each attester, the resource the attester had to attest or delegate, and his or her action.

URI
Request Parameters

You can customize the history log using the following parameters:

- **Scope**
  The `Scope` parameter specifies what part of the attestation history is displayed for the user. It can have the following three values:
  - **Global**
    The request returns information about all of the attestations that were previously executed. This is the default value.
  - **User**
    The request returns information about all of the attestations done for a specific user. If the `Scope` parameter has the value `User`, you must also specify the parameter `UserId`.
  - **Assignment**
    The request returns information about all of the actions that were performed on a specific attestation resource. If you use the `Assignment` value, you must specify the `LinkId` parameter. The `LinkId` parameter identifies a specific attestation resource. For example, it indicates the connection between a user and a role.

  **Note**
  If the `Scope` parameter is omitted in the URI, the system assumes that `Scope=Global`.

- **TaskType**
  Currently, this parameter has only one possible value: `Attestation`.

  **Note**
  If the `TaskType` parameter is omitted in the URI, the system assumes that `TaskType=Attestation`.

```plaintext
Parameter         = '?' Scope '& TaskType '& UserId '& LinkId
Scope             = 'Scope=' ScopeValue
TaskType          = 'TaskType=' Attestation
UserId            = 'UserId=' UserIdValue
LinkId            = 'LinkId=' LinkIdValue
ScopeValue        = Global | User | Assignment
UserIdValue       = Edm.Int32;
LinkIdValue       = Edm.Int32;
```

Request Example

```
[GET] idmrestapi/v2/service/TaskHistory
```

With this request, the response contains all of the information for the history of attestations.
Request Example with Global Scope

[GET] idmrestapi/v2/service/TaskHistory?Scope=Global&TaskType=Attestation

With this request, the response contains all of the information for the history of attestations.

Request Example with User Scope

[GET] idmrestapi/v2/service/TaskHistory?Scope=User&userId=16&TaskType=Attestation

With this request, the response contains all of the attestations done for the user with the ID 16. The name of this user is Julie Armstrong.

Request Example with Assignment Scope

[GET] idmrestapi/v2/service/TaskHistory?Scope=Assignment&LinkId=31&TaskType=Attestation

With this request, the response contains all the attestations done for the role Administrator and the user Julie Armstrong. The value 31 for the LinkId parameter indicates that the user Julie Armstrong has the Administrator role.

Response Types

Content-Type: application/json

HTTP Response Codes

- 200 - OK
- 401 - Unauthorized
  If the user is not authenticated.
- 403 - Forbidden
  If the user is not authorized to query for a history log.
- 405 - Method Not Allowed
  If the URI was not called with a GET request.
Response Example

```json
{
  d:
  {
    results:
    [2]
    0:
    {
      _metadata:
      {
        uri: "/idmrestapi/v2/service/TaskHistory(LINK_ID='31',OPERATION='Attestation: confirmed')"
        type: "TASKPROCESSING.TaskHistoryType"
      }
      LINK_ID: 31
      USER_DISPLAYNAME: "Julie Armstrong"
      USER_ID: 16
      ROLE_DISPLAYNAME: "Administrator"
      ROLE_ID: 26
      CONTEXT_DISPLAYNAME: null
      CONTEXT_ID: 0
      ASSIGNED_DATE: "/Date(1377561600000)/"
      VALID_FROM: null
      VALID_TO: null
      RESPONSIBLE_DISPLAYNAME: "Donna Moore"
      RESPONSIBLE_ID: 12
      OPERATION: "Attestation: confirmed"
      REASON: "Julie needs the Administrator role for some settings."
      AUDIT_DATE: "/Date(1379548800000)/"
    }
    1:
    {
      _metadata:
      {
        uri: "/idmrestapi/v2/service/TaskHistory(LINK_ID='32',OPERATION='Attestation: confirmed')"
        type: "TASKPROCESSING.TaskHistoryType"
      }
      LINK_ID: 32
      USER_DISPLAYNAME: "Michael Adams"
      USER_ID: 17
      ROLE_DISPLAYNAME: "Manager"
      ROLE_ID: 27
      CONTEXT_DISPLAYNAME: null
      CONTEXT_ID: 0
      ASSIGNED_DATE: "/Date(1377561600000)/"
      VALID_FROM: null
      VALID_TO: null
      RESPONSIBLE_DISPLAYNAME: "Donna Moore"
      RESPONSIBLE_ID: 12
      OPERATION: "Attestation: confirmed"
      REASON: "Michael is still a manager in the organization."
      AUDIT_DATE: "/Date(1378684800000)/"
    }
  }
}
```
11 Service Operations

The following sections provide an overview of the available service operations of the SAP SAP NetWeaver Identity Management OData service, which can be called on the service root URI.

11.1 Get Version Information

This service operation returns version information about the Identity Management REST service. This information contains a major version number and minor version number. The major version number is increased when an incompatible change was done. The minor number is increased for every release where compatible changes were done.

Clients can use this information to check if the major or minor version number is higher or equals to a version number that is required by a certain feature. See also Compatibility.

URI

```
(* [GET] *)
URI = './idmrestapi/v2/service/GetVersion';
```

Request Parameters

None.

Request Example

```
GET /idmrestapi/v2/service/GetVersion HTTP/1.1
```

Response Types

```
Content-Type: application/json
```
HTTP Response Codes

- 200 - OK

Response Example

```json
{
  "d":{
    "VersionInformation":{
      "MAJOR_VERSION":2,
      "MINOR_VERSION":0
    }
  }
}
```

Related Information

Compatibility [page 15]
Important Disclaimers and Legal Information

Coding Samples

Any software coding and/or code lines / strings ("Code") included in this documentation are only examples and are not intended to be used in a productive system environment. The Code is only intended to better explain and visualize the syntax and phrasing rules of certain coding. SAP does not warrant the correctness and completeness of the Code given herein, and SAP shall not be liable for errors or damages caused by the usage of the Code, unless damages were caused by SAP intentionally or by SAP’s gross negligence.

Accessibility

The information contained in the SAP documentation represents SAP’s current view of accessibility criteria as of the date of publication; it is in no way intended to be a binding guideline on how to ensure accessibility of software products. SAP in particular disclaims any liability in relation to this document. This disclaimer, however, does not apply in cases of willful misconduct or gross negligence of SAP. Furthermore, this document does not result in any direct or indirect contractual obligations of SAP.

Gender-Neutral Language

As far as possible, SAP documentation is gender neutral. Depending on the context, the reader is addressed directly with "you", or a gender-neutral noun (such as "sales person" or "working days") is used. If when referring to members of both sexes, however, the third-person singular cannot be avoided or a gender-neutral noun does not exist, SAP reserves the right to use the masculine form of the noun and pronoun. This is to ensure that the documentation remains comprehensible.

Internet Hyperlinks

The SAP documentation may contain hyperlinks to the Internet. These hyperlinks are intended to serve as a hint about where to find related information. SAP does not warrant the availability and correctness of this related information or the ability of this information to serve a particular purpose. SAP shall not be liable for any damages caused by the use of related information unless damages have been caused by SAP’s gross negligence or willful misconduct. All links are categorized for transparency (see: http://help.sap.com/disclaimer).