REST API Application Development

The SAP® Mobile Platform REST Services API enables standard HTTP client applications running in any platform to access SAP Mobile Platform REST services.

SAP Mobile Platform Server provides REST services so that applications can be built as any standard HTTP application to leverage SAP Mobile Platform Server for security and push features among others.

Build client applications using third-party (JavaScript framework and helper libraries) developer tools, native client libraries, or the libraries provided by SAP Mobile Platform OData SDK (Only iOS and Android platforms). SAP Mobile Platform provides management and monitoring of the applications, and support for native push notification such as Apple Push Notification service (APNS), BlackBerry Internet/Enterprise Service (BIS/BES), Google Cloud Messaging (GCM), Windows Notification Service (WNS), or Microsoft Push Notification Service (MPNS).

The client application should first register an application connection with device information, such as device type, password capability, and so on. After registering, the application can retrieve and update the application connection settings through the REST API. You can enable or disable the push notification only after registering.

**Note:** An application connection can be deleted through the REST API, as long as it is not in use. Any data that is stored by application in the custom string of the application connection properties is lost.

To initialize a client application, the application can download resources (such as metadata files, multimedia files, and so on.), using the resource bundles service.

After downloading resources, the application can access OData-compatible data sources through the proxy service, and receive native push notifications triggered by the gateway if push properties are configured and enabled.

This development approach supports:

- Registration (creating an application connection)
- Authentication
- Native push notification
- Configuration

For more information on supported REST API application development environment versions in SAP Mobile Platform 3.0, see [http://service.sap.com/pam](http://service.sap.com/pam).
Configuring the Application in the Management Cockpit

Configure the application settings in the Management Cockpit. These settings enable you to monitor and manage your applications.

Prerequisites

- Make sure SAP Mobile Platform Server is installed.
- Make sure the server is started.
- Launch the Management Cockpit.

Task

Defining Applications

Create a new native, hybrid, or Agentry application definition, which enables you to use Management Cockpit to manage the application.

1. In Management Cockpit, select Applications, and click New.
2. In the New Application window, enter:
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
</table>
| ID      | Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server. Reverse domain notation means reversing a registered domain name; for example, reverse domain notation for the object MyApp.sap.com is com.sap.MyApp. The application identifier:  
  • Must be unique.  
  • Must start with an alphabetic character.  
  • Can contain only alphanumeric characters, underscores (\_), and periods (\).  
  • Cannot include spaces.  
  • Can be up to 64 characters long.  
  Note: The keywords that are not allowed to be entered as application identifiers include: Admin, AdminData, Push, smp_cloud, resource, test-resources, resources, Scheduler, odata, applications, Connections, public. These keywords are case sensitive. Formatting guidelines:  
  • SAP recommends that application IDs contain a minimum of two periods (\.). For example, this ID is valid: com.sap.mobile.app1.  
  • Application IDs cannot start with a period (\.). For example, this ID is invalid: .com.sap.mobile.app1.  
  • Application IDs cannot include two consecutive periods (\.). For example, this ID is invalid: com..sap.mobile.app1. |
| Name    | Application name. The name:  
  • Can contain only alphanumeric characters, spaces, underscores (\_), and periods (\).  
  • Can be up to 80 characters long. |
| Vendor  | (Optional) Vendor who developed the application. The vendor name:  
  • Can contain only alphanumeric characters, spaces, underscores (\_), and periods (\).  
  • Can be up to 255 characters long. |
### Field | Value
--- | ---
Application type. | • Native – native applications, including Android, BlackBerry, iOS, Windows Mobile 8, and Windows 8.  
• Hybrid – Kapsel container-based applications.  
• Agentry – metadata-driven application. You can configure only one Agentry application per SAP Mobile Platform Server; after that, Agentry no longer appears as an option.

Description | (Optional) Short description of the application. The description:
- Can contain alphanumeric characters.
- Can contain most special characters, except percent signs (%) and ampersands (&).
- Can be up to 255 characters long.

3. Click **Save**. You see application-related tabs, such as Back End, Authentication, Push, and so forth. The tabs you see differ by application type. You are ready to configure the application, based on the application type.

**Note:** These tabs appear in Management Cockpit only after you define or select an application. The steps that follow assume you have selected the application, and are working through each of the relevant tabs for your selected application. The steps use a "navigational shorthand"—such as "select **Applications > Back End**"—to indicate the tab where tasks are performed, relative to that selected application, rather than repeating the entire navigation instruction.

### Defining Back-end Connections for Native and Hybrid Apps
Define back-end connections for the selected native or hybrid application. SAP Mobile Platform supports one primary endpoint per application ID. However, the administrator can create multiple secondary endpoints for other services used by the application; SAP Mobile Platform treats these additional endpoints as proxy connections.

1. From Management Cockpit, select the **Home** tab, and then **Configure Application**. Alternatively, select the **Applications** tab.
2. On the Applications tab, select one of the applications.

You see application-related tabs, such as Back End, Authentication, Push, and so forth. The tabs you see differ by application type. You are ready to configure the application, based on the application type.

**Note:** These tabs appear in Management Cockpit only after you define or select an application. The steps that follow assume you have selected the application, and are working through each of the relevant tabs for your selected application. The steps use a navigational shorthand—such as "select **Applications > Back End**"—to indicate the tab
where tasks are performed, relative to that selected application, rather than repeating the entire navigation instruction.

3. Enter values for the selected application:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Name</td>
<td>(Appears only when adding a connection under Back-End Connections.) Identifies the back-end connection by name. The connection name:</td>
</tr>
<tr>
<td></td>
<td>• Must be unique.</td>
</tr>
<tr>
<td></td>
<td>• Must start with an alphabetic character.</td>
</tr>
<tr>
<td></td>
<td>• Can contain only alphanumeric characters, underscores (_), and periods (.).</td>
</tr>
<tr>
<td></td>
<td>• Cannot include spaces.</td>
</tr>
<tr>
<td>Endpoint</td>
<td>The URL (back-end connection, or service document) the application uses to access business data on the back-end system or service. The service document URL is the document destination you assigned to the service in Gateway Management Cockpit. Include a trailing slash to avoid triggering a redirection of the URL, and losing important HTTP header details. This is especially important when configuring the application with security, such as SSOToken and Certificates, and when Rewrite URL is enabled. Typical format: <a href="http://host:port/gateway/odata/namespace/Connection_or_ServiceName.../">http://host:port/gateway/odata/namespace/Connection_or_ServiceName.../</a> Examples: <a href="http://testapp:65908/help/abc/appl/opg/sdata/TEST-FLIGHT/">http://testapp:65908/help/abc/appl/opg/sdata/TEST-FLIGHT/</a> <a href="http://srvc3333.xyz.com:30003/sap/opu/odata/RMTSAMPLE/">http://srvc3333.xyz.com:30003/sap/opu/odata/RMTSAMPLE/</a></td>
</tr>
<tr>
<td>Use System Proxy</td>
<td>(Optional) Whether to use system proxy settings in the SAP Mobile Platform props.ini file to access the back-end system. This setting is typically disabled, because most back-end systems can be accessed on the intranet without a proxy. Enable this setting only in unusual cases, where proxy settings are needed to access a remote back-end system outside of the network. When enabled, this particular connection is routed via the settings in props.ini file.</td>
</tr>
<tr>
<td>Rewrite URL</td>
<td>(Optional) Whether to mask the back-end URL with the equivalent SAP Mobile Platform Server URL. Enable this setting to ensure the client makes all requests via SAP Mobile Platform Server, and directly to the back end. Rewriting the URL also ensures that client applications need not do any additional steps to make requests to the back end via SAP Mobile Platform Server. If enabled, the back-end URL is rewritten with the SAP Mobile Platform Server URL. By default, this property is enabled.</td>
</tr>
</tbody>
</table>
### Field | Value
---|---
**Allow Anonymous Access** | (Optional) Whether to enable anonymous access, which means the user can access the application without entering a user name and password. However, the back-end system still requires login credentials for data access, whether it is a read-only user, or a back-end user with specific roles.
- If enabled and the back end requires it, enter the login credential values used to access the back-end system:
  - **User name** – supply the user name for the back-end system.
  - **Password** – (required if you set a user name) supply the password for the back-end system.
- If disabled (the default value) or the back end does not require it, you need not provide these credentials.

**Note:** If you use Allow Anonymous Access for a native OData application, do not also assign the No Authentication Challenge security profile to the application; anonymous OData requests are not sent, and Status code: 401 is reported.

**Maximum Connections** | The number of back-end connections that are available for connection pooling for this application. The larger the pool, the larger the number of possible parallel connections to this specific connection. The default and minimum is 500 connections. Factors to consider when resetting this property:
- The expected number of concurrent users of the application.
- The load that is acceptable to the back-end system.
- The load that the underlying hardware and network can handle.
Increase the maximum number of connections only if SAP Mobile Platform Server hardware can support the additional parallel connections, and if the underlying hardware and network infrastructure can handle it.

**Certificate Alias** | If the back-end system has a mutual SSL authentication requirement, supply the certificate alias name given to the private key and technical user certificate that is used to access the back-end system. The alias is located in `smp_keystore`. Otherwise, leave the entry blank.

4. **(Optional)** Under Back-end Connections, view additional connections, or add new connections.
   a) Click **New**, to add additional back-end connections in the server.
   b) Enter values for the new back-end connection, using the values shown above.
   c) Click **Save**. The new back-end connection is added to the list.

You can maintain the list of server-level back-end connections (including all the connections in SAP Mobile Platform Server), and of application-specific back-end connections. Application-specific back-end connections are the secondary connections that are enabled for an application; by default, no secondary connections are enabled. You must explicitly enable additional back-end connections for an application. Users who are registered to an application can access only these back-end connections. If a user attempts
to access a back-end connection (request-response) that is not enabled for an application, it is not allowed and a 403, Forbidden error is thrown.

5. Select Application-specific Connections from the drop-down to show the back-end connections that are enabled for the application.

Select Server-level Connections from the drop-down to show all available connections for the server. Use the checkbox to enable additional connections for the application.

**Note:**
- You can authenticate multiple back ends using various authentication provider options in the back-end security profile.
- If the back-end system issues a “302 Redirect” response, which means it is redirecting the request to a different URL, then you must also add the target URL to the list of application-specific connections.

---

**Defining Application Authentication**

Assign a security profile to the selected application. The security profile defines parameters that control how the server authenticates the user during onboarding, and request-response interactions with the back end.

**Prerequisites**

Configure security profiles for application authentication.

**Task**

Security profiles are made up of one or more authentication providers. These authentication providers can be shared across multiple security profiles, and can be modified in Management Cockpit. For more information on authentication providers, see *Authentication in SAP Mobile Platform*.

You can stack multiple providers to take advantage of features in the order you chose; the Control Flag must be set for each enabled security provider in the stack.

1. From Management Cockpit, select **Applications > Authentication**.
2. Click **Existing Profile**.

**Note:** You can also create a new profile.

3. Select a security profile name from the Name list.
   The name appears under Security Profile Properties, and the providers that are associated with the security profile appear under Authentication Providers.
Field | Value
--- | ---
Name | A unique name for the application authentication profile.
Check Impersonation | (Optional) In token-based authentication, whether to allow authentication to succeed when the user name presented cannot be matched against any of the user names validated in the login modules. By default the property is enabled, which prevents the user authentication from succeeding in this scenario.

5. Under Authentication Providers, you can select a security profile URL to view its settings. To change its settings, you must modify it using Settings > Security Profiles.

**Configuring Push on SAP Mobile Platform Server**

You must explicitly register the application connection using the Management Cockpit.

1. Start the Management Cockpit.
2. Select Applications, and click New.
3. In the New Application window, enter values.

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID</td>
<td>Unique identifier for the application, in reverse domain notation. This is the application or bundled identifier that the application developer assigns or generates during application development. The administrator uses the Application ID to register the application to SAP Mobile Platform Server, and the client application code uses the Application ID while sending requests to the server.</td>
</tr>
<tr>
<td>Name</td>
<td>Application name.</td>
</tr>
<tr>
<td>Vendor</td>
<td>(Optional) Vendor who developed the application.</td>
</tr>
<tr>
<td>Version</td>
<td>Application version. Currently, only version 1.0 is supported.</td>
</tr>
</tbody>
</table>
| Type | Application type.  
• Native – native iOS and Android applications.  
• Hybrid – container-based applications, such as Kapsel.  
• Agentry – metadata-driven applications, such as Agentry. Application configuration options differ depending on your selection. |
| Description | (Optional) Short description of the application. |

4. Click the Backend tab and configure the endpoint information.
5. Click the Push tab to configure the push settings.
   For Android GCM, see *Android Push Notifications*.
   For Apple APNS, see *Apple Push Notifications*
6. In the settings for your device, enable the app to receive push notifications.
**Android Push Notifications**
Configure Android push notifications for the selected application, to enable client applications to receive Google Cloud Messaging (GCM) notifications.

1. From Management Cockpit, select **Applications > Push**.
2. Under Android, enter the access key for API key. This is the access key you obtained for your Google API project (http://developer.android.com/google/gcm/gs.html).
3. Enter a value for Sender ID. This is the project identifier.
4. (Optional) Configure push notifications for each device type supported.

**Apple Push Notifications**
Configure Apple push notifications for the selected application, to enable client applications to receive APNS notifications.

1. From Management Cockpit, select **Applications > Push**.
2. Under Apple, select **APNS endpoint**. "None" is the default endpoint value for all the applications.
3. Select **Sandbox** to configure APNS in a development and testing environment, or **Production** to configure APNS in a production environment.
   a) Click **Browse** to navigate to the certificate file.
   b) Select the file, and click **Open**.
   c) Enter a valid password.

   **Note:** The default URL is for a production environment; for a development and testing environment, change the URL to gateway.sandbox.push.apple.com.

4. (Optional) Configure push notifications for each device type supported.

**Developing the Application**
Develop your HTTP client application to use the REST Services API to access SAP Mobile Platform REST services.

**Authentication Requests**
For all requests that require authentication, send the authentication information to SAP Mobile Platform. The credentials depend on the type of security configuration. Credentials should be provided in the header.

- **Basic authentication**
  The user name and password should be valid for the specified authentication URL.
- **HTTP Header Name: Authorization**
• HTTP Header Value: Basic <base64 encoded form of username:password>
• SAP SSO authentication
  The user name and password should be valid for the specified ticket-issuing system (TIS) URL.
• HTTP Header Name: Authorization
• HTTP Header Value: Basic <base64 encoded form of username:password>
• SiteMinder SSO authentication (Client acquires SSO Token)
  • HTTP Header Name: <value provided for 'Client HTTP Values To Send' in the security configuration>
  • HTTP Header Value: actual SMSESSION token
• SiteMinder SSO authentication (SAP Mobile Platform acquires SSO token)
  The user name and password should be valid for the specified ticket-issuing system (SiteMinder server) URL.
  • HTTP Header Name: Authorization
  • HTTP Header Value: Basic <base64 encoded form of username:password>
• Certificate authentication
  Prepare a client certificate and get it signed by certification authority (CA) certificate of the server. The client certificate should be trusted by SAP gateway or any other EIS. The certificate then be used to register the client and perform the request-response with the server.

**Creating an Application Connection**

You must explicitly register the application connection using SAP Mobile Platform.

You can specify customized application properties for the client with the request. Provide the application connection ID, X-SMP-APPCID, using an explicit request header or a cookie. If the value is missing, SAP Mobile Platform generates a universally unique ID (UUID), which is communicated to the device through the response cookie X-SMP-APPCID.

Create an anonymous or authenticated application connection by issuing a POST request to this URL, including the application connection properties:

http[s]://<host:port>[/public/]/odata/applications/{latest|v1}/ {appid}/Connections

The URL contains these components:

• **host** – the host is defined by host name and should match with the domain registered with SAP Mobile Platform. If the requested domain name does not match, default domain is used.
• **port** – the port for listening to OData-based requests. By default the port number is 8080.
• **public** – if included, an anonymous connection is allowed.
• **odata/applications/** – refers to the OData services associated with the application resources.
• {latest|v1} – version of the service document.
• appid – name of the application.
• Connections – name of the OData collection.

Application connection properties are optional. You can create an application connection without including any application properties.

DeviceInfo is an application connection property that you may set. Valid values for DeviceType are:

• Android – Android devices.
• iPhone – Apple iPhone.
• iPad – Apple iPad.
• iPod – Apple iPod.
• iOS – iOS devices.
• Blackberry – Blackberry devices.
• Windows – includes desktop or servers with Windows OS, such as Windows XP, Windows Vista, Windows 7, and Windows Server series OS.
• WinPhone8 – includes Windows mobile.
• Windows8 – includes Windows desktop version.

Specifying any other value for DeviceType returns a value "Unknown" in the DeviceType column.

Example of creating an application connection

Request:

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <content type="application/xml">
    <m:properties>
      <d:AndroidGcmRegistrationId>398123745023</d:AndroidGcmRegistrationId>
    </m:properties>
  </content>
</entry>
```

Response

```xml
<?xml version="1.0" encoding="utf-8"?>
  <id>https://<application URL>/odata/applications/latest/e2eTest/</id>
</entry>
```
Connections('4891dd0f-0735-47cc-a599-76bf8a16d457')</id>
<title type="text" />
<updated>2012-10-19T09:25Z</updated>
<author>

</author>
<link rel="edit" title="Connection" href="Connections('4891dd0f-0735-47cc-a599-76bf8a16d457')" />
<content type="application/xml">
  <m:properties>
    <d:ETag>2012-10-19 14:35:24.0</d:ETag>
    <d:ApplicationConnectionId>4891dd0f-0735-47cc-a599-76bf8a16d457</d:ApplicationConnectionId>
    <d:AndroidGcmRegistrationId>398123745023</d:AndroidGcmRegistrationId>
    <d:ApnsDeviceToken />
    <d:ApplicationVersion>1.0</d:ApplicationVersion>
    <d:BlackberryDevicePin m:null="true" />
    <d:BlackberryBESListenerPort m:type="Edm.Int32">0</d:BlackberryBESListenerPort>
    <d:BlackberryPushAppID m:null="true" />
    <d:BlackberryPushBaseUrl m:null="true" />
    <d:BlackberryPushListenerPort m:type="Edm.Int32">0</d:BlackberryPushListenerPort>
    <d:BlackberryListenerType m:type="Edm.Int32">0</d:BlackberryListenerType>
    <d:CustomizationBundleId />
    <d:CustomCustom1 />
    <d:CustomCustom2 />
    <d:CustomCustom3 />
    <d:CustomCustom4 />
    <d:DeviceModel m:null="true" />
    <d:DeviceType>Unknown</d:DeviceType>
    <d:DeviceSubType m:null="true" />
    <d:DevicePhoneNumber m:null="true" />
    <d:DeviceIMSI m:null="true" />
    <d>PasswordPolicyEnabled m:type="Edm.Boolean">false</d>PasswordPolicyEnabled>
    <d>PasswordPolicyDefaultPasswordAllowed m:type="Edm.Boolean">false</d>PasswordPolicyDefaultPasswordAllowed>
    <d>PasswordPolicyMinLength m:type="Edm.Int32">0</d>PasswordPolicyMinLength>
    <d>PasswordPolicyDigitRequired m:type="Edm.Boolean">false</d>PasswordPolicyDigitRequired>
    <d>PasswordPolicyUpperCaseRequired m:type="Edm.Boolean">false</d>PasswordPolicyUpperCaseRequired>
    <d>PasswordPolicyLowerCaseRequired m:type="Edm.Boolean">false</d>PasswordPolicyLowerCaseRequired>
  </m:properties>
</content>
CORS Support
Cross-domain HTTP requests are requests for resources from a different domain than the
domain of the resource making the request. Cross-Origin Resource Sharing (CORS)
mechanism provides a way for web servers to support cross-site access controls, which enable
secure cross-site data transfers.

Managing Application Settings
Application settings describe the application connection details such as application ID,
security configuration, and customization resource.

Getting Application Settings
You can retrieve application connection settings for the device application instance by issuing
the GET method.

You can retrieve application settings by either explicitly specifying the application connection
ID, or by having the application connection ID determined from the call context (that is, from
either the X-SMP-APPCID cookie or X-SMP-APPCID HTTP header, if specified). On the
first call, you can simplify your client application code by having the application connection
ID determined from the call context, since you have not yet received an application connection
ID.

If you supply an application connection ID, perform an HTTP GET request at:
http[s]://<host:port>/{public/}odata/applications/{latest|v1}/
{appid}/Connections('{appcid}')

Response
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
<title type="text" />
<author><name /></author>
<link rel="edit" title="Connections" href="Connections('c9d8a9da-9f36-4ae5-9da5-37d6d90483b5')" />
<content type="application/xml">
  <m:properties>
    <d:ApplicationConnectionId>c9d8a9da-9f36-4ae5-9da5-37d6d90483b5</d:ApplicationConnectionId>
    <d:AndroidGcmRegistrationId m:null="true" />
    <d:AndroidGcmSenderId m:null="true" />
    <d:ApnsPushEnable m:type="Edm.Boolean">true</d:ApnsPushEnable>
    <d:ApnsDeviceToken m:null="true" />
    <d:ApplicationVersion m:null="true" />
    <d:BlackberryDevicePin>00000000</d:BlackberryDevicePin>
    <d:BlackberryBESListenerPort m:type="Edm.Int32">5011</d:BlackberryBESListenerPort>
    <d:BlackberryPushAppID m:null="true" />
    <d:BlackberryPushBaseURL m:null="true" />
    <d:BlackberryPushListenerPort m:type="Edm.Int32">0</d:BlackberryPushListenerPort>
    <d:BlackberryListenerType m:type="Edm.Int32">0</d:BlackberryListenerType>
    <d:CustomCustom1>custom1</d:CustomCustom1>
    <d:CustomCustom2 m:null="true" />
    <d:CustomCustom3 m:null="true" />
    <d:CustomCustom4 m:null="true" />
    <d:DeviceModel m:null="true" />
    <d:DeviceType>Unknown</d:DeviceType>
    <d:DeviceSubType m:null="true" />
    <d:DevicePhoneNumber>12345678901</d:DevicePhoneNumber>
    <d:DeviceImsi m:null="true" />
    <d:PasswordPolicyMinLength m:type="Edm.Int32">8</d:PasswordPolicyMinLength>
    <d:PasswordPolicyMinLength>
      <d:PasswordPolicyDigitRequired m:type="Edm.Boolean">false</d:PasswordPolicyDigitRequired>
You can also retrieve a property value by appending the property name in the URL. For example, to retrieve the ClientLogLevel property value, enter:

http[s]://<host:port>/[public/]odata/applications/{v1|latest}/ {appid}/Connections('{appcid}')/ClientLogLevel

**Retrieving Changed Settings and Connections Metadata**

You can conditionally retrieve only the changed settings and connections metadata.

To retrieve the changed application settings information:

http[s]://<host:port>/[public/]odata/applications/{latest|v1}/ {appID}/Connections('{appcid}')?If-None-Match="${ETag}"

The ${ETag} part of the URL is a version identifier included in the response of the GET method. If the ETag value of the current application settings is the same as the value included in the request, a status code 304 without a response body is returned to the client to indicate that there are no application setting changes.

**Note:** The query function is not currently supported.

To retrieve the metadata document of the SAP Mobile Platform Server:

http[s]://<host:port>/[public/]odata/applications/{latest|v1}/ {appID}/$metadata

To retrieve the service document of the SAP Mobile Platform Server:

http[s]://<host:port>/[public/]odata/applications/{latest|v1}/ {appID}
Setting or Updating Application Settings

Set or update application settings by issuing a PUT request.

Use the PUT method to update the application settings with all the properties in the request.

```
PUT http[s]://<host:port>/[public/]odata/applications/{latest|v1}/
{appid}/Connections('{appcid}')
```

Use the PUT method to update specified properties. Any properties you do not specify retain their current values.

```
http[s]://<host:port>/[public/]odata/applications/{latest|v1}/
{appid}/Connections('{appcid}')
```

The URL contains these components:

- **host:port** – Hostname and port number of SAP Mobile Platform.
- **public** – If included, an anonymous security configuration is used.
- **odata/applications/** – Refers to the OData services associated to the application resources.
- **{latest|v1}** – The version of the service document.
- **appid** – The name of the application.
- **Connections** – The name of the OData collection.
- **appcid** – The application connection ID of the application instance that is interacting with the service.

Native Push Notification for a Back End

The SAP Mobile Platform supports native notification for device platforms.

The SAP Mobile Platform uses the native notification mechanisms provided by individual device platforms such as APNS, GCM, BIS/BES, WNS, and MPNS to send notifications. Back end systems use the 'Push' REST service to notify the SAP Mobile Platform about any notification message it has to send to the devices.

Delivery address URL format is:

```
http[s]://<host:port>/Notification/<registration id>
```

where:

- **host:port/Notification** can be received from the proxy push endpoint.
- **registration ID** is sent to the device when the user registers and connects to the application from the device.

The notification data can also be sent using URL arguments.
Notification Data Sent Through HTTP Headers

Notification data can be sent by the back end as a generic HTTP headers or as device platform-specific HTTP headers.

The notification URL is:

http[s]://<host:port/Notification>/<registration id>

**Note:** Applications built in SAP Mobile Platform 3.0 and later should adopt the header format X-SMP-XXX. To maintain backward compatibility, applications built in earlier versions can continue to use the header format X-SUP-XXX. However, X-SUP-XXX headers will be removed future releases.

- **Generic header**
  The generic HTTP header is used in the HTTP request to send any notification type such as APNS, GCM, Blackberry, or WNS.

  Header format for notification data in SAP Mobile Platform 3.x and later:

  `<X-SMP-DATA>`

- **APNS-specific headers**
  Use these APNS-specific HTTP headers to send APNS notifications via SAP Mobile Platform:

<table>
<thead>
<tr>
<th>Header Structure (SAP Mobile Platform and later)</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT&gt;</code></td>
<td>A JSON document. You can use this header or other individual headers listed in this table.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-BODY&gt;</code></td>
<td>Text of the alert message.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-ACTION-LOC-KEY&gt;</code></td>
<td>If a string is specified, this header shows an alert with two buttons: Close and View. iOS uses the string as a key to get a localized string for the correct button title instead of View. If the value is null, the system shows an alert. Clicking <strong>OK</strong> dismisses the alert.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-LOC-KEY&gt;</code></td>
<td>Key to an alert-message string in a Localizable.strings file for the current localization.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-LOC-ARGS&gt;</code></td>
<td>Variable string values to appear in place of the format specifiers in loc-key.</td>
</tr>
</tbody>
</table>
### Header Structure (SAP Mobile Platform and later)

<table>
<thead>
<tr>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-LAUNCH-IMAGE&gt;</code></td>
</tr>
<tr>
<td>File name of an image file in the application bundle. It may include the extension. Used as the launch image when you tap the action button or move the action slider. If this property is not specified, the system uses one of the following:</td>
</tr>
<tr>
<td>• The previous snapshot</td>
</tr>
<tr>
<td>• The image identified by the <code>UILaunchImage-File</code> key in the Info.plist file of the application</td>
</tr>
<tr>
<td>• The <code>Default.png</code></td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-BADGE&gt;</code></td>
</tr>
<tr>
<td>Number that appears as the badge on the application icon.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-SOUND&gt;</code></td>
</tr>
<tr>
<td>Name of the sound file in the application bundle.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-DATA&gt;</code></td>
</tr>
<tr>
<td>Custom payload data values. These values must use the JSON-structured and primitive types, such as dictionary (object), array, string, number, and Boolean.</td>
</tr>
</tbody>
</table>


- **GCM-specific headers**

Use these GCM-specific HTTP headers to send GCM notifications:

<table>
<thead>
<tr>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-GCM-COLLAPSEKEY&gt;</code></td>
</tr>
<tr>
<td>An arbitrary string (such as “Updates Available”) that collapses a group of like messages when the device is offline, so that only the last message is sent to the client.</td>
</tr>
</tbody>
</table>

**Note:** If you do not include this header, the default value “Updates Available, is used.

| `<X-SMP-GCM-DATA>`                                                         |
| Payload data, expressed as parameters prefixed with data and suffixed as the key. |

| `<X-SMP-GCM-DELAYWHILEIDLE>`                                               |
| (Optional) Represented as 1 or true for true, any other value for false, which is the default value. |
Header Structure (SAP Mobile Platform and later) | Consists of
--- | ---
`<X-SMP-GCM-TIMETOLIVE>` | Time (in seconds) that the message remains available on GCM storage if the device is offline.


- **BES/BIS-specific header**
  Use the BlackBerry-specific HTTP header to send BES/BIS notifications:
  `<x-sup-rim-data>` or `<X-SMP-RIM-DATA>`

- **WNS specific header**
  Use these HTTP headers to send Windows 8 desktop and tablet application notifications:

<table>
<thead>
<tr>
<th>Header Structure (SAP Mobile Platform and later)</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-WNS-DATA&gt;</code></td>
<td>Send payload data to the device as raw notification. Payload data may also be a binary data encoded as a Base64-encoded string. Size should not exceed 5KB.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-WNS-ALERT&gt;</code></td>
<td>Text string of the notification, as Tile and Toast notifications.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-WNS-BADGE&gt;</code></td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
</tbody>
</table>

- **MPNS (Notification for Windows Phone)**
  Use these Windows Phone-specific HTTP headers to send MPNS notifications:

<table>
<thead>
<tr>
<th>Request Header Structure</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-MPNS-DATA&gt;</code></td>
<td>Send payload data to device as raw notification. Payload data may also be a binary data encoded as a Base64-encoded string. String length should not exceed more than 2900 characters.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-MPNS-ALERT&gt;</code></td>
<td>Text string of the notification, as Tile and Toast notifications.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-MPNS-BADGE&gt;</code></td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
</tbody>
</table>
SAP Gateway Notification Support
There are no specific handling requirements for sending notifications on the SAP gateway side. SAP Mobile Platform sends notifications using gateway-specific headers.

The SAP Mobile Platform identifies the device type, based on the device type converts the gateway notification headers into the third-party notification context data for APNS, GCM or BES/BIS, WNS, and MPNS.

Note: Non-SAP gateway back ends also use the headers listed below to send generic notifications. In this scenario, the backend is unaware of the device platform.

SAP gateway-specific headers that are handled by SAP Mobile Platform for sending notifications:

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;x-sap-poke-title&gt;</td>
<td>Text of the alert message</td>
</tr>
<tr>
<td>&lt;x-sap-poke-entriesofinterest&gt;</td>
<td>Number that appears as the badge on the application icon</td>
</tr>
<tr>
<td>&lt;x-sap-poke-data&gt;</td>
<td>Custom payload data values. These values must use the JSON structured and</td>
</tr>
<tr>
<td></td>
<td>primitive types such as dictionary (object), array, string, number, and</td>
</tr>
<tr>
<td></td>
<td>boolean</td>
</tr>
</tbody>
</table>

1. APNS
The SAP Mobile Platform converts the gateway notification headers into APNS notifications:

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;x-sap-poke-title&gt;</td>
<td>Text of the alert message</td>
</tr>
<tr>
<td>&lt;x-sap-poke-entriesofinterest&gt;</td>
<td>Number that appears as the badge on the application icon</td>
</tr>
<tr>
<td>&lt;x-sap-poke-data&gt;</td>
<td>Custom payload data values. These values must use the JSON structured and</td>
</tr>
<tr>
<td></td>
<td>primitive types such as dictionary (object), array, string, number, and</td>
</tr>
<tr>
<td></td>
<td>boolean</td>
</tr>
</tbody>
</table>

2. GCM
The SAP Mobile Platform converts the gateway notification headers into GCM notifications:
### Header Structure

<table>
<thead>
<tr>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;x-sap-poke-title&gt;</td>
</tr>
<tr>
<td>An arbitrary string (such as &quot;Updates Available&quot;) collapses a group of</td>
</tr>
<tr>
<td>like messages when the device is offline, so that only the last message</td>
</tr>
<tr>
<td>is sent to the client</td>
</tr>
<tr>
<td>&lt;x-sap-poke-data&gt;</td>
</tr>
<tr>
<td>Payload data. Size should not exceed 4KB</td>
</tr>
</tbody>
</table>

3. **BIS/BES**

The SAP Mobile Platform converts the gateway notification headers into BIS/BES notifications:

<table>
<thead>
<tr>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;x-sap-poke-data&gt;</td>
</tr>
<tr>
<td>BES/BIS notification data</td>
</tr>
</tbody>
</table>

4. **WNS**

The SAP Mobile Platform converts the gateway notification headers into WNS notifications:

<table>
<thead>
<tr>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;x-sap-poke-title&gt;</td>
</tr>
<tr>
<td>Text of the alert message to be shown on</td>
</tr>
<tr>
<td>the Tile and Toast notifications</td>
</tr>
<tr>
<td>&lt;x-sap-poke-entriesofinterest&gt;</td>
</tr>
<tr>
<td>Number that appears as the badge on the</td>
</tr>
<tr>
<td>application icon</td>
</tr>
<tr>
<td>&lt;x-sap-poke-data&gt;</td>
</tr>
<tr>
<td>Custom payload data to be sent to the</td>
</tr>
<tr>
<td>device as a raw notification</td>
</tr>
</tbody>
</table>

### Notification Sent in URL Format

The notification data can also be sent by using the REST client using URL arguments as part of the SAP Mobile Platform push endpoint, or as the delivery address URL. The use of embedded arguments in the URL is not supported.

All URL arguments (zero to many) are optional. The arguments are converted into APNS/GCM/BES/WNS/MPNS notifications as explained:

```
http[s]://<host:port/Notification>/<application connection ID>?
alert=<alert>&badge=<badge>&sound=<sound>&data=<data in text format>
```

- **APNS**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>Text of the alert message.</td>
</tr>
</tbody>
</table>
### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>badge</td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
<tr>
<td>sound</td>
<td>Name of the sound file in application bundle.</td>
</tr>
<tr>
<td>data</td>
<td>Custom payload data values. These values must use the JSON-structured and primitive types, such as dictionary (object), array, string, number, and boolean.</td>
</tr>
</tbody>
</table>

#### GCM

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>An arbitrary string (such as &quot;Updates Available&quot;) that collapses a group of like messages when the device is offline, so that only the last message is sent to the client.</td>
</tr>
<tr>
<td>data</td>
<td>Payload data, expressed as parameters prefixed with data and suffixed as the key.</td>
</tr>
</tbody>
</table>

#### BIS/BES

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data</td>
<td>Notification data</td>
</tr>
<tr>
<td>alert</td>
<td>Text of the alert message</td>
</tr>
<tr>
<td>badge</td>
<td>Number that appears as the badge on the application icon</td>
</tr>
</tbody>
</table>

#### WNS

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>The text of the alert message to be sent as a Tile notification</td>
</tr>
<tr>
<td>badge</td>
<td>Number that appears as the badge on the application icon</td>
</tr>
<tr>
<td>data</td>
<td>Payload data to be sent</td>
</tr>
</tbody>
</table>

#### MPNS (Notification for Windows Phone)
### REST API Application Development

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alert</td>
<td>The text of the alert message to be sent as a Tile notification</td>
</tr>
<tr>
<td>badge</td>
<td>Number that appears as the badge on the application icon</td>
</tr>
<tr>
<td>data</td>
<td>Payload data to be sent</td>
</tr>
</tbody>
</table>

Based on the data send either in headers or in the URL, corresponding notification is sent to the device:

<table>
<thead>
<tr>
<th>Header</th>
<th>Notification</th>
<th>Tile Notification</th>
<th>Toast Notification</th>
<th>Raw Notification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alert</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Badge</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td></td>
</tr>
<tr>
<td>Data</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
<td></td>
</tr>
</tbody>
</table>

### Registering the Client for Native Push Notification

Enable native push notifications and register your application to receive push notifications.

**Prerequisites**
- Configure the registration ID before registering the client for native notifications.
- Configure the application to send push notifications.

### Registering the Android Client

Register and enable your Android device clients to receive push notifications.

**Prerequisites**
- (Administrator) Configure the application for push notification in Management Cockpit by specifying the sender ID and API key.
- Specify the device type should be specified during application connection and registration.
- HTTP header must include X-SMP-APP CID and Authorization headers.

**Task**

1. If `AndroidGcmPushEnabled` is enabled, sender ID is sent in the response. On successful onboarding of client, the response indicates the GCM push is enabled.
2. If GCM is enabled and the sender ID is available, client uses that sender ID to register itself with GCM and get its unique GCM registration ID.

3. Use the PUT method in the URL, along with the registration ID:

   http://<host:port>/odata/applications/\{latest\|v1\}/\{appid\}/Connections/\('\{appcid\}'\)
   
   Method : PUT
   
   HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "MERGE"
   
   Body:
   
   
   <m:properties>
   
   <d:AndroidGcmRegistrationId>{GCM registration ID}</d:AndroidGcmRegistrationId>
   
   </m:properties>
   
   </entry>`

**Registering the BlackBerry Client**

Register and enable your BlackBerry device clients to receive push notifications.

1. Configure push notification in Management Cockpit. Specify the device type during application connection and registration, and ensure that the HTTP header also includes the X-SMP-APPCID and Authorization headers.

   **Note:** To configure push notifications for BIS, you must import BIS certificate into the smp_keystore.jks and keystore files in the configuration folder of the SAP Mobile Platform Server.

2. Enable push notifications in the application:
   
   a) Update the application connection settings with the BES/BIS registration ID.
   
   a) **Implement the BlackberryPushListenerPort and BlackberryDevicePin properties in your application.**

   **Http payload to update the blackberry (BES) device PIN and push port:**

   `<?xml version="1.0" encoding="utf-8"?>
   
   
   <m:properties>
   
   <d:BlackberryDevicePin/>
   <d:BlackberryBESListenerPort><XXXX/></d:BlackberryBESListenerPort>
   
   </m:properties>
   
   </entry>`
Http payload to update the blackberry (BIS) device PIN and push port:

```xml
<?xml version="1.0" encoding="utf-8"?>
  <m:properties>
    <d:BlackberryPushEnabled>true</d:BlackberryPushEnabled>
    <d:BlackberryDevicePin> </d:BlackberryDevicePin>
    <d:BlackberryPushAppID></d:BlackberryPushAppID>
    <d:BlackberryPushBaseURL> </d:BlackberryPushBaseURL>
    <d:BlackberryPushListenerPort><XXXX></d:BlackberryPushListenerPort>
  </m:properties>
</entry>
```

Registering the iOS Client
Register and enable your iOS device clients to receive push notifications.

1. Configure push notification in Management Cockpit. Specify the device type during application connection and registration.
2. Enable push notifications in the application:
   a) To receive the device token, implement the `application:didRegisterForRemoteNotificationsWithDeviceToken:` method in your application delegate.
   b) Update the `ApnsDeviceToken` and `DeviceType` properties via a PUT request. The HTTP header must also include X-SMP-APPCID and Authorization headers.

```xml
https://<host:port>/odata/applications/{latest|v1}/{appid}/Connections/('{appcid}')
Method : PUT
HTTP Headers : "Content-Type" = "application/atom+xml"

Body:
<?xml version='1.0' encoding='utf-8'?>
  <id>https://{application URL}/odata/applications/{latest|v1}/e2eTest/Connections('32552613-470f-45e0-8acc-b7d73d501682')</id>
  <content type="application/xml">
    <m:properties>
      <d:ApnsDeviceToken>{APNS device token received by the application from APNS}</d:ApnsDeviceToken>
      <d:DeviceType>iPhone</d:DeviceType>
    </m:properties>
  </content>
</entry>
```
Registering the Windows 8 (Desktop and Tablet) Client
Register and enable your Windows 8 (desktop and tablet) devices to receive push notifications.

Prerequisites

- Configure Windows push notification in Management Cockpit.
- The HTTP header must also include X-SMP-APPCID and Authorization headers.

Task

1. To obtain the channel URI, register the application with WNS. See Push notification overview (Windows Store apps) on the Windows Dev Center Web site.

2. Check the WnsPushEnable value returned from the registration, and continue with the rest of the WNS or notification registration processing only if the value is true. Set the WnsChannelURI value received from the application.

3. Update the application connection settings with the registration ID:

   ```
   https://host:port/odata/applications/{latest|v1}/{appid}/Connections/(''{appcid}'')
   Method : PUT
   HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "PUT"
   Body:
   <content type="application/xml">
     <m:properties>
       <d:WnsChannelURI>{WNS Channel URI}</d:WnsChannelURI>
     </m:properties>
   </content>
   </entry>
   ```

Registering the MPNS Client
Register and enable your Windows phone 8 to receive push notifications.

Prerequisites

- Administrator configures the application for push notification in Management Cockpit.
- The HTTP header must also include X-SMP-APPCID and Authorization headers.
Task

1. To obtain the channel URI, register the application with MPNS. See *Push notifications for Windows Phone* on the Windows Phone Dev center Web site.

2. Check the `MpnsPushEnable` value returned from the registration, and continue with the rest of the MPNS or notification registration processing only if the value is true. Set the `MpnsChannelURI` value received from the application.

3. Using the `ApplicationConnection ID` ({appcid}) returned from the SAP Mobile Platform registration call (in either the X-SMP-APPCID HTTP header or the `ApplicationConnectionId` property), the application should update the `MpnsChannelURI` property for the application connection for your application by using the Channel URI returned by the application:

   ```
   https://host:port/odata/applications/{latest|v1/}{appid}/
   Connections/(''{appcid}'')
   Method : PUT
   HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "PUT"
   Body:
   <content type="application/xml">
   <m:properties>
   <d:MpnsChannelURI>{MPNS Channel URI}</d:MpnsChannelURI>
   </m:properties>
   </content>
   </entry>
   ```

Retrieving a Customization Resource Bundle

Retrieve a customization resource bundle by issuing a GET method.

Application developers can customize and retrieve the resource bundles during development.

Issue the GET method to the following URL:

```
http[s]://<host:port>[/public/] bundles/{appid}/
[{resourceBundlename:resourceBundleVersion}]
```

If the `{resourceBundlename:resourceBundleVersion}` is specified in the URL, the specified resource bundle is returned in the response body as a stream, otherwise, the resource bundle that is bound to the application is returned. Extension of the resource bundle is provided in the response header X-BUNDLE-EXTENSION.

If the resource bundle is not found in SAP Mobile Platform, error code 404 is returned. You cannot issue other HTTP methods (PUT/POST/DELETE) at the above URL.
Accessing a Service Through a Proxy URL

Access your backend or internet-based service through a proxy URL. The URL supports read (HTTP GET), create (HTTP POST), update (HTTP PUT), and delete (HTTP DELETE).

You can specify the customized application properties for the client with the request. Provide the application connection ID, X-SMP-APPCID by using an explicit request header or a cookie.

Perform an HTTP request at the following URL:

http[s]://<host:port>/[public/]/ApplicationName/{connectionName}

Note:

- Ensure that all the URLs to be proxied are whitelisted.
- If the session is not handled by the client implicitly, you need to handle it explicitly in the code.

- **HTTP GET** – Invoke GET request to retrieve the data from the back end through SAP Mobile Platform.

  http[s]://<host:port>/[public/]/ApplicationName/[Collection]

  Example:

  Request header
  
  X-SMP-APPCID : <Application connection Id received in the response of the onboarding xml>
  Authorization : <Base64 format of username/password>
  Content-type : <application/atom+xml>
  X-CSRF-Token: <"Fetch">

  Response header
  
  Response code 200 ok
  //Response header contains requested data from back end with the below mentioned response headers:
  x-csrf-token: "value"
  content-encoding
  Content-Type
  Content-Length

- **HTTP POST** – Use POST to request the server to accept the data included in the request message body.

  http[s]://<host:port>/[public/]/ ApplicationName/[Collection]

  Example:

  Request header
  
  X-SMP-APPCID : <Application connection Id received in the response of the onboarding xml>
Content-Type : application/atom+xml
X-Requested-With : XMLHttpRequest
X-CSRF-TOKEN : <value of x-csrf-token is received as part of GET which needs to be passed here>
Authorisation : <Base 64 encoded value of Authorisation>

Response header
201 //on successful creation.

• **HTTP PUT** – Use PUT to update the entry in the back end.

http[s]://<host:port>/<public>/ApplicationName/[Collection]/('EntryID')

Example:

Request header
X-SMP-APPCID : <Application connection Id received in the response of the onboarding xml>
Content-Type : application/atom+xml
X-CSRF-TOKEN : <value of x-csrf-token is received as part of GET which needs to be passed here>
Authorisation : < Base 64 encoded value of Authorisation>
X-Requested-With : XMLHttpRequest

Response header
204 //on successful update.

• **HTTP DELETE** – Use DELETE to the delete an entry in back end.

http[s]://<host:port>/<public>/ApplicationName/[Collection]/('EntryID')

Example:

Request header
X-SMP-APPCID : <Application connection Id received in the response of the onboarding xml>
Authorization : <Base64 format of username/password>
Content-type : <application/atom+xml>
X-CSRF-TOKEN : <value of x-csrf-token is received as part of GET which needs to be passed here>
X-Requested-With : XMLHttpRequest

Response header
204 //on successful deletion.

**Note:** No information is returned in case of DELETE request.
Reference

Describes REST API resources.

Note: The examples in the reference section use basic authentication, which you can see in the HTTP header, for example Authorization: Basic Base64-encoded string.

HTTP Headers and Cookies

Use HTTP headers and cookies to retrieve application connection information.

Note: In SAP Mobile Platform 3.0, applications should adopt the header format X-SMP-XXX. To maintain backward compatibility, the applications built in earlier version continue to use the header format X-SUP-XXX. However, X-SUP-XXX headers will be removed from future releases.

Cookies are returned by servers in the HTTP response header (Set-Cookie header) and included by the HTTP client (for example, a browser) in the subsequent HTTP request header (cookie header). Supported headers are:

- **X-SMP-APPCID** – communicates the application connection ID as that is generated by the application from which the request originates, or as issued by the server using the X-SMP-APPCID cookie. The X-SMP-APPCID value has a maximum length of 128 characters. If an onboarding request does not include X-SMP-APPCID, the server automatically generates a value. The client or application should send this value for all subsequent requests. If the X-SMP-APPCID value is not sent from the client, all subsequent request-response interactions result in an **403 Forbidden** error.

  X-SMP-APPCID is received as a cookie, but the client returns it to the server either as a cookie or as a header.

- **X-SMP-BACKEND-URL** – application can provide the back-end request URL via an X-SMP-BACKEND-URL header while sending the request to SAP Mobile Platform Server. The server forwards the request to the URL provided in the X-SMP-BACKEND-URL.

  If the X-SMP-BACKEND-URL header is present in the request, URL rewrite is disabled.

  Note: To avoid errors, ensure that the URL is whitelisted in the server.

- **X-SMP-REQUESTID** – communicates the request ID as generated by the application from which the request originates, or as issued by the server that is using the X-SMP-REQUESTID cookie.
Application Connection Properties

Describes application connection properties, and indicates whether the properties are read-only or Nullable from the HTTP client.

**Note:** If you attempt to modify a read-only property, the client application throws the following exception: HTTP 403 - The property "XXX" cannot be updated by a client application.

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETag</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ApplicationConnectionId</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

**Table 1. Application Connection Properties: Uncategorized**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>AndroidGcmPushEnabled</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>AndroidGcmRegistrationId</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>AndroidGcmSenderId</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Table 2. Application Connection Properties: Android Push**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApnsPushEnable</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ApnsDeviceToken</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Table 3. Application Connection Properties: Apple Push**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomizationBundleId</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ApplicationVersion</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ClientSdkVersion</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

**Table 4. Application Connection Properties: Application Settings**
### Table 5. Application Connection Properties: BlackBerry Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlackberryPushEnabled</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>BlackberryDevicePin</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>BlackberryBESListenerPort</td>
<td>Int32</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

### Table 6. Application Connection Properties: Windows Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>WnsChannelURI</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>WnsPushEnable</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

### Table 7. Application Connection Properties: MPNS Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MpnsChannelURI</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>MpnsPushEnable</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

### Table 8. Application Connection Properties: Capabilities

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CapabilitiesPasswordPolicy</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

### Table 9. Application Connection Properties: Custom Settings

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomCustom1</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom2</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom3</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom4</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Table 10. Application Connection Properties: Device Information

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeviceModel</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceType</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceSubType</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DevicePhoneNumber</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceIMSI</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Table 11. Application Connection Properties: Password Policy

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PasswordPolicyEnabled</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyDefaultPasswordAllowed</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyMinLength</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyDigitRequired</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyUpperCaseRequired</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyLowerCaseRequired</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicySpecialRequired</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyExpiresInNDays</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyMinUniqueChars</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyLockTimeout</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyRetryLimit</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 12. Application Connection Properties: Proxy

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProxyApplicationEndpoint</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ProxyPushEndpoint</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Proxy Responses
Proxy responses include all the cookies and headers from the proxied backend.

Application Connections
Methods for creating, updating, or reading application connections.

**Note:** Application connection service is implemented as an OData service and therefore follows OData standards.

Service Document
Get the service document for the application connection. Retrieving the service document allows the client to discover the capabilities and locations of the available collections.

**Syntax**
Perform an HTTP GET request at the following URL:

```
http[s]://<host:port>/[public/]odata/applications/\{v1|latest\}/\{appid\}
```

**Parameters**
- **appid** – The application ID that uniquely identifies the application.

**Returns**
Returns a 200 OK status code if successful, and a service document in the response body.

**Examples**
- **Service document** – HTTP request header:

```
GET /odata/applications/v1/com.sap.myapp HTTP/1.1
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic REVWMDAwMTppbml0aWFs
```

HTTP response header:
```
HTTP/1.1 200 OK
Content-Type: application/xml; charset=utf-8
```
The HTTP response body is a service document that includes two collections named "Connections" and "Endpoints".

```xml
<?xml version='1.0' encoding='utf-8'?>
<service xmlns="http://www.w3.org/2007/app"
    xmlns:atom="http://www.w3.org/2005/Atom"
    xmlns:app="http://www.w3.org/2007/app">
    <workspace>
        <atom:title>Default</atom:title>
        <collection href="Connections">
            <atom:title>Connections</atom:title>
        </collection>
        <collection href="Endpoints">
            <atom:title>Endpoints</atom:title>
        </collection>
    </workspace>
</service>
```

**Metadata**

Get the metadata document that includes the metadata for the application connection settings and proxy endpoints.

Metadata documents are based on the OData standard and required for implementing application connection services.

**Syntax**

Perform an HTTP GET request at the following URL:

```
http[s]://<host:port>/[public/]odata/applications/{v1|latest}/
{appid}/$metadata
```

**Parameters**

- `appid` – The application ID that uniquely identifies the application.

**Returns**

If successful, returns a 200 OK status code and a metadata document in the response body.

**Examples**

- **Get Metadata** – HTTP request header:

  ```
  GET /odata/applications/v1/com.sap.myapp/$metadata HTTP/1.1
  Host: smpserver:8080
  Connection: Keep-Alive
  ```
REST API Application Development

User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic REVWMDAwMTppbml0aWFs

HTTP response header:
HTTP/1.1 200 OK
Content-Type: application/xml;charset=utf-8
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: X-SMP-SESSID=97ts80gwhxkc;Path=/
Set-Cookie: X-SMP-APPCID=b6d50e93-bcaa-439d-9741-660a3cb56771
DataServiceVersion: 1.0
Date: Tue, 14 Aug 2012 21:32:34 GMT
Transfer-Encoding: chunked

HTTP response body (metadata):
<?xml version="1.0" encoding="utf-8"?>
<EntityType Name="Connection">
  <Key>
    <PropertyRef Name="ApplicationConnectionId"></PropertyRef>
  </Key>
  <Property Name="ETag" Type="Edm.String" Nullable="false" sup:ReadOnly="true"/>
  <Property Name="ApplicationConnectionId" Type="Edm.String" Nullable="false" sup:ReadOnly="true"/>
  <Property Name="AndroidGcmPushEnabled" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="AndroidGcmRegistrationId" Type="Edm.String" Nullable="true" sup:ReadOnly="false"/>
  <Property Name="AndroidGcmSenderId" Type="Edm.String" Nullable="true" sup:ReadOnly="true"/>
  <Property Name="ApnsPushEnable" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="ApnsDeviceToken" Type="Edm.String" Nullable="true" sup:ReadOnly="false"/>
  <Property Name="ApplicationVersion" Type="Edm.String" Nullable="true" sup:ReadOnly="false"/>
  <Property Name="BlackberryPushEnabled" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="BlackberryDevicePin" Type="Edm.String" Nullable="true" sup:ReadOnly="false"/>
  <Property Name="BlackberryBESListenerPort" Type="Edm.Int32" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="BlackberryPushAppID" Type="Edm.String" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="BlackberryPushBaseURL" Type="Edm.String" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="BlackberryPushListenerPort" Type="Edm.Int32" Nullable="false" sup:ReadOnly="false"/>
  <Property Name="BlackberryListenerType" Type="Edm.Int32" Nullable="false" sup:ReadOnly="false"/>
</EntityType>
</Schema>
</edmx:DataServices>
</edmx:Edmx>
<Property Name="ConnectionLogLevel" Type="Edm.String" Nullable="true" sup:ReadOnly="true"></Property>
<Property Name="CustomizationBundleId" Type="Edm.String" Nullable="true" sup:ReadOnly="true"></Property>
<Property Name="CustomCustom1" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="CustomCustom2" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="CustomCustom3" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="CustomCustom4" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="DeviceModel" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="DeviceType" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="DeviceSubType" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="DevicePhoneNumber" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="DeviceIMSI" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="MpnsChannelURI" Type="Edm.String" Nullable="true" sup:ReadOnly="false"></Property>
<Property Name="MpnsPushEnable" Type="Edm.Boolean" Nullable="false" sup:readOnly="false"></Property>
<Property Name="PasswordPolicyEnabled" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyDefaultPasswordAllowed" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyMinLength" Type="Edm.Int32" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyDigitRequired" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyUpperRequired" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyLowerRequired" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicySpecialRequired" Type="Edm.Boolean" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyExpiresInNDays" Type="Edm.Int32" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyMinUniqueChars" Type="Edm.Int32" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyLockTimeout" Type="Edm.Int32" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="PasswordPolicyRetryLimit" Type="Edm.Int32" Nullable="false" sup:ReadOnly="true"></Property>
<Property Name="ProxyApplicationEndpoint" Type="Edm.String" Nullable="true" sup:ReadOnly="true"></Property>
Create Application Connection

Create an application connection and initially set the application connection settings. Because all application connection settings are optional, the minimal body contains no properties at all. SAP Mobile Platform populates default values as needed.

**Syntax**

Perform an HTTP POST request at the following URL:

http[s]://<host:port>/[public/]/odata/applications/{v1|latest}/ {appid}/Connections

**Parameters**

- **appid** – The application ID that uniquely identifies the application.

  **Note:** If an application is configured for anonymous access using Management Cockpit, the registration is successful irrespective of wrong or no credentials present in the authorization header.

**Returns**

If successful, a 201 Created status code is returned, and the new application connection settings are included in the response body.

**Examples**

- **Create application connection** – Request minimal body:

```xml
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
```
REST API Application Development

metadata">
<title type="text"/>
<updated>2014-061-15T02:23:29Z</updated>
<author>
<name/>
</author>
<content type="application/xml">
<m:properties>
<d:DeviceType>iPhone</d:DeviceType>
<d:DeviceModel m:null="true" />
<d:ApnsDeviceToken m:null="false">18A4813FB9E6393065BFEDADCD68173782A42599F3C9E2BF14F990F2D9F096</d:ApnsDeviceToken>
</m:properties>
</content>
</entry>

Response body:
/id>http://10.53.138.236:8080/odata/applications/latest/docTest/Connections('a21eb32b-0eb5-4dbe-baaa-8e5ca079176e')</id>
<title type="text" />
<updated>2014-01-06T09:19:18Z</updated>
<author>
<name />
</author>
<link rel="edit" title="Connection" href="Connections('a21eb32b-0eb5-4dbe-baaa-8e5ca079176e')" />
<content type="application/xml">
<m:properties>
<d:ETag>2014-01-06 14:49:18.0</d:ETag>
<d:ApplicationConnectionId>a21eb32b-0eb5-4dbe-baaa-8e5ca079176e</d:ApplicationConnectionId>
<d:AndroidGcmRegistrationId m:null="true" />
<d:ApnsPushEnable m:type="Edm.Boolean">false</d:ApnsPushEnable>
<d:ApnsDeviceToken>18A4813FB9E6393065BFEDADCD68173782A42599F3C9E2BF14F990F2D9F096</d:ApnsDeviceToken>
<d:ApplicationVersion>1.0</d:ApplicationVersion>
<d:BlackberryDevicePin>2100000A</d:BlackberryDevicePin>
<d:BlackberryBESListenerPort m:type="Edm.Int32">0</d:BlackberryBESListenerPort>
<d:BlackberryPushappId m:null="true" />
Get Application Settings
Get the application settings.
**Syntax**

Perform an HTTP GET request at the following URL:

```
http[s]://<host:port>/[public/]odata/applications/{v1|latest}/
{appid}/Connections('appcid')
```

**Parameters**

- **appid** – The application ID that uniquely identifies the application.
- **appcid** – The application connection ID of the application instance interacting with the service.

**Returns**

If successful, returns a 200 OK status code and an HTTP response body with the application settings.

**Examples**

- **Get application settings – HTTP request header:**

  ```
  GET /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-
bcaa-439d-9741-660a3cb56771') HTTP/1.1
  Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
  Host: smpserver:8080
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
  Authorization: Basic <XXXX>
  ```

  **HTTP response header:**

  ```
  HTTP/1.1 200 OK
  Content-Type: application/atom+xml;charset=utf-8
  Expires: Thu, 01 Jan 1970 00:00:00 GMT
  Set-Cookie: X-SMP-APPCID=<XXXX>
  DataServiceVersion: 1.0
  Date: Mon, 13 Aug 2012 22:56:50 GMT
  Transfer-Encoding: chunked
  ```

  **HTTP response body:**

  ```
  <?xml version="1.0" encoding="UTF-8"?>
  <!DOCTYPE atom:entry SYSTEM "http://docs.oasis-open.org/atom/atom盆地.dtd">
  <atom:entry xmlns:atom="http://www.w3.org/2005/Atom"
              xmlns:atom="http://www.w3.org/2005/Atom">
    <atom:title type="text"/>
    <atom:author>
      <atom:name/></atom:author>
    <atom:link title="Connection" href="/odata/applications/latest/G3T/Connections/<XXX>'"/>
  </atom:entry>
  ```
dataservices/scheme" term="applications.Connection"/>
<content type="application/xml">
<m:properties>
  <d:ETag>2013-11-07 14:44:43.0</d:ETag>
  <d:ApplicationConnectionId>xxxx</d:ApplicationConnectionId>
  <d:AndroidGcmRegistrationId m:null="true"/>
  <d:AndroidGcmSenderId/>
  <d:ApnsPushEnable m:type="Edm.Boolean">true</d:ApnsPushEnable>
  <d:ApnsDeviceToken m:null="true"/>
  <d:ApplicationVersion>1.0</d:ApplicationVersion>
  <d:BlackberryBESListenerPort m:type="Edm.Int32">0</d:BlackberryBESListenerPort>
  <d:BlackberryPushAppID m:null="true"/>
  <d:BlackberryPushBaseURL m:null="true"/>
  <d:BlackberryPushListenerPort m:type="Edm.Int32">0</d:BlackberryPushListenerPort>
  <d:BlackberryListenerType m:type="Edm.Int32">0</d:BlackberryListenerType>
  <d:ConnectionLogLevel>NONE</d:ConnectionLogLevel>
  <d:CustomizationBundleId m:null="true"/>
  <d:DeviceModel m:null="true"/>
  <d:DeviceType>Unknown</d:DeviceType>
  <d:DeviceSubType m:null="true"/>
  <d:DevicePhoneNumber m:null="true"/>
  <d:DeviceIMSI m:null="true"/>
  <d:MpnsChannelURI m:null="true"/>
  <d:MpnsPushEnable m:type="Edm.Boolean">true</d:MpnsPushEnable>
  <d>PasswordPolicyEnabled m:type="Edm.Boolean">false</d>PasswordPolicyEnabled>
  <d>PasswordPolicyDefaultPasswordAllowed m:type="Edm.Boolean">false</d>PasswordPolicyDefaultPasswordAllowed>
  <d>PasswordPolicyMinLength m:type="Edm.Int32">8</d>PasswordPolicyMinLength>
  <d>PasswordPolicyDigitRequired m:type="Edm.Boolean">false</d>PasswordPolicyDigitRequired>
  <d>PasswordPolicyUpperRequired m:type="Edm.Boolean">false</d>PasswordPolicyUpperRequired>
  <d>PasswordPolicyLowerRequired m:type="Edm.Boolean">false</d>PasswordPolicyLowerRequired>
  <d>PasswordPolicySpecialRequired m:type="Edm.Boolean">false</d>PasswordPolicySpecialRequired>
  <d>PasswordPolicyExpiresInNDays m:type="Edm.Int32">0</d>PasswordPolicyExpiresInNDays>
  <d>PasswordPolicyMinUniqueChars m:type="Edm.Int32">0</d>PasswordPolicyMinUniqueChars>
  <d>PasswordPolicyLockTimeout m:type="Edm.Int32">0</d>PasswordPolicyLockTimeout>
Get Application Settings (Property)
Get the specific property value for a property from the application settings.

Syntax
Perform an HTTP GET request at the following URL:

http://<host:port>[/public/]odata/applications/{v1|latest}/{appid}/Connections('{registrationID}')/<property-name>

Parameters

- **appid** – The application ID that uniquely identifies the application.
- **registrationID** – The registration ID of the application instance that is interacting with the service.
- **property-name** – The property-name can be appended to the URL to retrieve the value of a specific property.

Returns
If successful, returns a 200 OK status code and an HTTP response body with the application settings.

Examples

- **Get the DeviceType property** – HTTP request:

  GET /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-bcaa-439d-9741-660a3cb56771')/DeviceType HTTP/1.1
  Cookie: X-SMP-APPCID=b6d50e93-bcaa-439d-9741-660a3cb56771; X-SMP-SESSID=97ts80gwhxkc
  Host: smpserver:8080
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
  Authorization: Basic REVWMDAwMTppbml0aWFs

  HTTP response header:
HTTP/1.1 200 OK
Content-Type: application/xml;charset=utf-8
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: X-SMP-APPCID=<XXXX>
DataServiceVersion: 1.0
Date: Mon, 13 Aug 2012 23:00:27 GMT
Transfer-Encoding: chunked

HTTP response body:
<?xml version='1.0' encoding='utf-8'?><d:DeviceType>iPhone</d:DeviceType>

**Update Application Settings (PUT)**

Update the application settings with all the properties in the request.

Any properties that you do not specify (and that can be changed) are set to the default value.

**Syntax**

PUT http[s]://<host:port>/[public/]odata/applications/{v1|latest}/
{appid}/Connections('{registrationID}"

**Parameters**

- **appid** – The application ID that uniquely identifies the application.
- **registrationID** – The registrationID of the application instance that is interacting with the service.

**Returns**

When processing a PUT request, SAP Mobile Platform returns a 200 status code to indicate success, and there is no response body.

If you have not explicitly registered the client, PUT request returns a 404 status code.

**Examples**

- **Update application settings (PUT)** – HTTP request header:

  PUT /odata/applications/v1/com.sap.myapp/Connections('<XXXX>')
  HTTP/1.1
  Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
  Content-Length: 4744
  Content-Type: application/atom+xml; charset=UTF-8
  Host: smpserver:8080
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
  Authorization: Basic <XXXX>

  [PUT HTTP request message body]

  HTTP response header:
Delete Application Connection
Delete an application connection.

Syntax
Perform an HTTP DELETE request at the following URL:
http[s]://<host:port>/[public/]odata/applications/{v1|latest}/
{appid}/Connections('{appcid}')

Parameters
• appid – The application ID that uniquely identifies the application.
• appcid – The application connection ID of the application instance interacting with the service.

Returns
If successful, returns a 200 OK status code.
If you never explicitly registered the client, returns a 404 status code.

Examples
• Delete application connection – HTTP request:
  DELETE /odata/applications/v1/com.sap.myapp/
  Connections('b6d50e93-bcaa-439d-9741-660a3cb56771') HTTP/1.1
  Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
  Host: smpserver:8080
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
  Authorization: Basic <XXXX>

  HTTP response:
  HTTP/1.1 200 OK
  Expires: Thu, 01 Jan 1970 00:00:00 GMT
  Set-Cookie: X-SMP-APPCID=<XXXX>
  DataServiceVersion: 1.0
  Date: Mon, 13 Aug 2012 23:19:42 GMT
  Content-Length: 0
Error Code and Message Format

The server returns different formats for error codes and messages according to different "Accept" values in request headers.

<table>
<thead>
<tr>
<th>Type and Format</th>
<th>Accept Header Values</th>
<th>Sample Response Body</th>
</tr>
</thead>
<tbody>
<tr>
<td>XML</td>
<td>application/xml, application/xhtml+xml, application/atom+xml</td>
<td>&lt;html&gt;&lt;head&gt;&lt;title&gt;&quot;message string&quot;&lt;/title&gt;&lt;/head&gt;&lt;body&gt;&lt;h1&gt;&quot;error code&quot; - &quot;error string&quot;.&lt;/h1&gt;&quot;&lt;p&gt;&lt;b&gt;message string&lt;/b&gt; &lt;u&gt;error string&lt;/u&gt;&lt;/p&gt;&lt;p&gt;&lt;b&gt;description&lt;/b&gt; &lt;u&gt;error message&lt;/u&gt;&lt;/p&gt;&lt;h3&gt;&quot;text string&quot;&lt;/h3&gt;&lt;/body&gt;&lt;/html&gt;</td>
</tr>
<tr>
<td>JSON</td>
<td>application/json, text/json</td>
<td>{ &quot;error&quot;: {&quot;code&quot;: &quot;403&quot;, &quot;message&quot;: {&quot;lang&quot;: &quot;en-US&quot;, &quot;value&quot;: &quot;some specific error text string&quot; } } }</td>
</tr>
<tr>
<td>TEXT</td>
<td>text/html, text/plain</td>
<td>&quot;some specific error text string&quot;</td>
</tr>
</tbody>
</table>

Note: If the Accept header does not include any of these data types, the response body is null.

Download Application Customization Resource Bundle

Download the customization resource bundle.

Note: If you add multiple resource bundles and no resource is specified in the URL, the one indicated as default considered as the default resource bundle.

Syntax

Perform an HTTP GET request at the following URL:

http[s]://<host:port>/[public/] bundles/{appid}/{{resourceBundleName:Version}}

Parameters

- resourceBundleName – Optionally specifies a resource bundle to be returned.
- Version – Version of the resource bundle.

Returns

If you specify the {resourceBundleName} in the URL, the specified resource bundle is returned in the response body as a stream. Otherwise, the resource bundle that is bound to the application is returned.
If the resource bundle is not found, error code 404 is returned. You cannot issue other HTTP methods (PUT/POST/DELETE) at the URL labels parameters shown in syntax.

**Examples**

- **Download** – HTTP request:

  ```
  GET /bundles/com.sap.myapp/MyApp:1.0 HTTP/1.1
  Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
  Host: smpserver:8080
  Connection: Keep-Alive
  User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
  Authorization: Basic <XXXX>
  
  HTTP response:
  HTTP/1.1 200 OK
  Expires: Thu, 01 Jan 1970 00:00:00 GMT
  Set-Cookie: X-SMP-APPCID=<XXXX>
  Transfer-Encoding: chunked
  
  [resource bundle content]
  ```

**Proxy Connections**

Methods for accessing proxy connections.

**Access External Service**

SAP Mobile Platform Server can also be used to access OData services that are provided by any external OData provider. The proxy URL supports read (HTTP GET), create (HTTP POST), update (HTTP PUT), and delete (HTTP DELETE).

**Syntax**

Perform an HTTP request at the following URL:

`http[s]://<host:port>[/public/]/[connectionName]`

**Parameters**

- `connectionName` – connection name of the whitelisted URL

**Note:** If an application is configured for anonymous access, the request-response is made using the same user credentials provided in "Allow anonymous connections" field for defining the backend connection in Management Cockpit.

**Returns**

If successful, returns a response from the back end and a response body.
Examples

• **Access external service** – HTTP request header:

```plaintext
GET /com.sap.myapp HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic <XXXX>
```

HTTP response header:

```plaintext
HTTP/1.1 200 OK
Content-Type: application/atomsvc+xml
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: X-SMP-APPCID=<XXXX>
serviceversion: 2.0
server: SAP NetWeaver Application Server / ABAP 731
data: some data
Content-Length: 2651
```

**CORS Enabled Browser-Based Applications**

Cross-Origin Resource Sharing (CORS) is a specification that allows scripts from one domain to make requests to another domain.

When a browser-based application, such as a JavaScript or jQuery application, sends a request to a domain other than the one it is hosted in, it is called a cross domain request. All valid CORS requests are accompanied by an origin header, which is added automatically by the browser. SAP Mobile Platform can handle all the received CORS requests.

The response headers that are added to all valid CORS requests received by SAP Mobile Platform include:

- **Access-Control-Allow-Origin** – origin value as specified in the request header.
- **Access-Control-Allow-Credentials** – when set to true, enables credential requests and requests accompanied with cookies.
- **Access-Control-Expose-Headers** – to include any custom header in the application, the list of exposed headers must be set as the value in this header.

**Note:** If the backend does not explicitly specify **Access-Control-Expose-Headers**, SAP Mobile Platform adds all response headers (except for simple HTTP headers and set-cookie header) to this header value, allowing the browser application to access all the headers from the backend response.

In case of a pre-flight (HTTP OPTIONS) request, SAP Mobile Platform adds these headers to the response:

- **Access-Control-Allow-Methods** – Based on the request URL:
  - POST for onboarding URL
• GET, PUT, DELETE for application settings URL
• GET for resource bundle URL
• Value of Access-Control-Request-Method header for all other URLs
• **Access-Control-Allow-Headers** – Access-Control-Request-Headers header value as specified in request.
• **Access-Control-Max-Age** – the time period for which the browser caches the results of the pre-flight request is by default, 3600 seconds.

**Browser Restrictions with CORS**

• Internet Explorer versions 9 and earlier do not support CORS-enabled browser-based applications.
• Safari supports Document Object Model (DOM) parsing with restrictions.
Index

A
APNS 25
Apple 25
applications
  configuring applications 2
  creating an application definition 2
authentication
  application, defining 7

B
BlackBerry devices 24
browser-based applications 48

C
configuring applications 2
CORS 48
creating an application definition 2
creating application endpoint URL 4
creating back-end connection 4
customization 46

D
defining
  application authentication 7
downloading 46

E
error code 46

G
error message 46

G
gateway notification 20

I
iOS devices 25

N
native push notifications 25, 26

R
resource bundles 46

S
Safari 48

W
windows devices 26

X
X-SMP-APPCID 24