# Content

1. **SAP Cloud Platform Mobile Service for Development and Operations** .................................................. 3
   1.1 Feature Scope. .................................................................................................................. 4
       About this Document. ........................................................................................................... 6
   1.2 What’s New. .................................................................................................................... 19
   1.3 Get Started. ..................................................................................................................... 21
       Enable Developer Accounts. ............................................................................................. 22
       Set Up Customer Accounts. ............................................................................................... 23
       Define Customer Account Types. ...................................................................................... 26
       Set Up Customer Accounts for Agentry. .......................................................................... 27
   1.4 Administration. ................................................................................................................ 28
       SAP Cloud Platform Mobile Service for Development and Operations Overview. ............. 30
       User Interface. .................................................................................................................. 41
       Application Administration. .............................................................................................. 42
       System Administration. ...................................................................................................... 177
       Security Administration. ................................................................................................... 182
       Maintenance Mode. .......................................................................................................... 221
       Troubleshooting: Common Issues. .................................................................................... 221
   1.5 Development. .................................................................................................................. 224
       REST API Application Development Overview. .............................................................. 224
   1.6 Migration. ......................................................................................................................... 411
       Migrating to SAP Cloud Platform Mobile Service for Development and Operations. .......... 412
   1.7 Glossary. ......................................................................................................................... 413
1 SAP Cloud Platform Mobile Service for Development and Operations

SAP Cloud Platform mobile service for development and operations is an open, standards-based cloud platform that enables simple mobile application development, configuration, and management.
1.1 Feature Scope

Summarizes the core features included in SAP Cloud Platform mobile service for development and operations.
SAP Cloud Platform Mobile Service for Development and Operations

**Work Offline**
When offline, switch to the local data source, which is kept current via OData synchronization features and open standards.

**Push**
Send updates and notifications from the backend data source to mobile apps. Use the Push Desk to send notifications to users of a specific app.

**SAP Fiori Client**
SAP Fiori Client is a native mobile application runtime container that enables you to access SAP Fiori Launchpad in online mode.

**App Development Support**
Supports development for the following app types:
- Native OData apps
- Hybrid apps
- Agentry (metadata) apps
- Web apps

**Security**
Choose the authentication types that suit your data needs.

**App Management**
Define, manage, and monitor your native, hybrid, Agentry, and Web apps through their entire lifecycles.

**App Updates**
Intelligent update mechanisms keep your apps and app configurations up-to-date through optimized downloads, and allow you to maintain multiple app versions in the field.

**Build or Extend**
Extend existing on-premise solutions, or use SAP Cloud Platform to build a fully cloud-hosted solution that uses HANA as the back end, the Java runtime, the HTML5 app, and other mobile app tools.

**Usage Reporting**
View app-specific usage statistics and reports.

**SAP Content to Go**
Share data from Fiori desktop apps to iPhones in a mobile-friendly format, either online or offline. When online, dynamic updates provide the most up-to-date business data.
1.1.1 About this Document

This document defines the functional scope of SAP Cloud Platform mobile service for development and operations. Further restrictions may apply based on your license agreement with SAP. Functions and capabilities described in the documentation of this product may exceed the functional scope of the product to explain the integration with other SAP products, which must be licensed separately.

1.1.2 Feature Descriptions

Core features in SAP Cloud Platform mobile service for development and operations. Available in English (EN), Simplified Chinese (ZH-CN), Japanese (ja-JP), and Korean (ko-KO).

Work Offline [page 7]
Users can seamlessly work with their apps, even if no network is available. The secure local data source is kept current via OData synchronization features and open standards. Administrators can configure offline settings, and monitor offline statistics.

Push [page 7]
Send updates and notifications from the back-end data source to mobile apps. Use the push desk to send ad-hoc notifications to users of a specific app.

SAP Fiori Client [page 9]
SAP Fiori Client is a native mobile application that easily consumes Fiori-based applications like the SAP Fiori Launchpad. It provides additional supportability features, Fiori-specific caching, access to device features, an integrated attachment viewer, and a better user experience than a mobile browser for this specific use case.

App Development Support [page 9]
Supports development for multiple app types.

Security [page 11]
Choose the authentication types that suit your data protection needs. Secure your system landscape by authenticating application users, establishing SSO connections to back-end systems, and protecting data and other resources.

App Management [page 13]
Define, manage, and monitor your native, hybrid, Agentry, and Web apps through their entire life cycles.

App Updates [page 15]
Intelligent update mechanisms keep your apps and app configurations up-to-date through optimized downloads, and allow you to maintain multiple app versions in the field.

Usage Reporting [page 16]
View app-specific usage statistics and reports.

SAP Content to Go [page 18]
Share data from virtually any back-end to iPhones in a mobile-friendly format, either online or offline. When online, dynamic updates provide the most up-to-date business data available.

Build or Extend [page 18]
Extend existing on-premise solutions, or use SAP Cloud Platform to build a fully cloud-hosted solution that uses HANA as the back end, the Java Runtime, HTML5 app, and mobile app tools.

### 1.1.2.1 Work Offline

Users can seamlessly work with their apps, even if no network is available. The secure local data source is kept current via OData synchronization features and open standards. Administrators can configure offline settings, and monitor offline statistics.

**Table 1: Offline Features**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>View offline configuration settings</td>
<td>View offline configuration settings in Mobile Service for Development and Operations cockpit.</td>
</tr>
<tr>
<td>View usage statistics for offline OData applications</td>
<td>Administrators can view request and response-time usage statistics for offline OData applications in Mobile Service for Development and Operations cockpit. Statistics are gathered for offline data store operations such as <code>build</code>, <code>refresh</code>, and <code>flush</code>.</td>
</tr>
<tr>
<td>Offline data security</td>
<td>The local data storage used for offline access is secured on the device. When the Offline Store Upload API is implemented, users can securely upload local database files to the server.</td>
</tr>
</tbody>
</table>

### 1.1.2.2 Push

Send updates and notifications from the back-end data source to mobile apps. Use the push desk to send ad-hoc notifications to users of a specific app.

**Table 2: Push Notification Features**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native push notifications</td>
<td>Use native push capabilities provided by Apple, Google, and Windows.</td>
</tr>
<tr>
<td>Custom push provider.</td>
<td>Configure applications to receive push notifications via private push vendors. For example, replace the Google Cloud Messaging (GCM) push service with a local vendor service. Configure the custom push provider for specific device types.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Push desk notification</td>
<td>(Native and hybrid apps only) Send an ad-hoc push notification from Mobile Service for Development and Operations cockpit to all users of a specific application or to a filtered list. For example, inform app users about an event, such as upcoming maintenance, or a new feature or promotion. The Notification User role is required.</td>
</tr>
<tr>
<td>View push-enabled devices for a selected application</td>
<td>In the cockpit, view a list of devices for which push is enabled.</td>
</tr>
<tr>
<td>Preconfigured push notifications</td>
<td>(SAP Content to Go and SAP Fiori Client) When enabled, preconfigured push settings that come with App Store and Google Play versions of an app are used to push notifications.</td>
</tr>
<tr>
<td>Push notification statistics</td>
<td>View push notification statistics for push enabled apps as part of usage reporting. You can view notification details, notifications by applications, and notifications by operating system.</td>
</tr>
<tr>
<td>Capability-based push support (iOS)</td>
<td>Registration services support capability handling and device type (form factor). Send push notifications to devices with a particular capability, rather than to individual applications. Use the Push API to enable, and the capability and form factor parameters to identify capabilities and device types.</td>
</tr>
<tr>
<td>Note</td>
<td>Hybrid apps do not support device-type (form factor), with the exception of Fiori Clients downloaded from the App Store, and custom Fiori clients.</td>
</tr>
<tr>
<td>Actionable push support (iOS)</td>
<td>Push notifications to users, enabling them to take action without changing focus. Use REST API headers and custom parameters to provide non-SAP gateway notification support (Category and Content Available).</td>
</tr>
<tr>
<td>Enhanced badge handling for capability-based push (iOS)</td>
<td>For capability-based push, the back end can send the badge number per capability. Use REST Services to enable.</td>
</tr>
<tr>
<td>Support canonical IDs for Google Cloud Messaging (GCM)</td>
<td>Prevents the client from receiving duplicate push messages.</td>
</tr>
<tr>
<td>Push API features</td>
<td>The push API includes JSON payload handling features for Android, Apple, and Windows, enabling you to map JSON parameter values to platform custom values.</td>
</tr>
</tbody>
</table>
1.1.2.3 SAP Fiori Client

SAP Fiori Client is a native mobile application that easily consumes Fiori-based applications like the SAP Fiori Launchpad. It provides additional supportability features, Fiori-specific caching, access to device features, an integrated attachment viewer, and a better user experience than a mobile browser for this specific use case.

Table 3: SAP Fiori Client Features

<table>
<thead>
<tr>
<th>Feature (by role)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Administrators</strong></td>
<td></td>
</tr>
<tr>
<td>Additional benefits when SAP Fiori Client</td>
<td>In the cockpit, define the application as a hybrid application.</td>
</tr>
<tr>
<td>connects to the Fiori front-end server</td>
<td></td>
</tr>
<tr>
<td>through SAP Cloud Platform mobile service</td>
<td>Control access to specific native device capabilities.</td>
</tr>
<tr>
<td>for development and operations</td>
<td></td>
</tr>
<tr>
<td><strong>End Users</strong></td>
<td></td>
</tr>
<tr>
<td>Additional native capabilities for Fiori</td>
<td>SAP Fiori apps running within Fiori Client can access native capabilities</td>
</tr>
<tr>
<td>apps beyond what a typical browser</td>
<td>such as such as camera, barcode scanner, push notification, and geolocation.</td>
</tr>
<tr>
<td>provides</td>
<td></td>
</tr>
<tr>
<td><strong>Better management of the local cache</strong></td>
<td>Reliable caching mechanism for SAP Fiori Client assets when new versions of</td>
</tr>
<tr>
<td></td>
<td>the application are released by the Fiori front-end server.</td>
</tr>
<tr>
<td><strong>Developers</strong></td>
<td></td>
</tr>
<tr>
<td>Custom SAP Fiori client (for iOS, Android,</td>
<td>Developers use the SAP Mobile Platform SDK to adjust the branding, add</td>
</tr>
<tr>
<td>and Windows)</td>
<td>support for custom authentication schemes, or add additional SAP, partner,</td>
</tr>
<tr>
<td></td>
<td>or open source plugins.</td>
</tr>
</tbody>
</table>

1.1.2.4 App Development Support

Supports development for multiple app types.

- Native
- Hybrid
- Agentry (metadata)
- Web

Table 4: App Development Support Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>REST services</td>
<td>Enables standard HTTP client applications running on any platform to leverag</td>
</tr>
<tr>
<td></td>
<td>e SAP Cloud Platform mobile service for development and operations for</td>
</tr>
<tr>
<td></td>
<td>security and push features.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Logout services</td>
<td>Enables users to terminate an active session from the client. The service does not require authentication, but does require a session cookie to identify the session to terminate. Use the logout service API to enable.</td>
</tr>
</tbody>
</table>
| Document services      | The SAP Cloud Platform document service is an implementation of the Content Management Interoperability Services (CMIS) standard, which allows you to create an application-specific repository. Use the document service API to enable. Features include:  
  - The repository can be used to store usage logs. To ensure privacy, the user must give consent before usage logs can be stored.  
  - You can access the document service repository through the management cockpit without registering. |
| Storage services       | Provides persistent storage that is separate from back-end storage. Developers can use REST services to store device application configuration data on the server. Administrators manage storage from the cockpit.  
  Storage service is replacing the original onboarding service to better serve a microservices-based architecture. Application-, user-, and device-level configuration information is now handled by separate storage services. |
| Client Resource service| As an application developer, you can use REST services to download resources from the server.                                                                 |
| Client Log Upload service| As an application developer, you can use REST services to upload client logs in the server.                                                                 |
| Sample OData service   | Use the sample OData service for development and testing. Manage the OData service from the cockpit. Administrators can view the root service and metadata URLs, and generate sample sales orders and purchase orders for multiple entity sets. You can also view the data for each entity in a separate text file, and reset the sample data. |
1.1.2.5 Security

Choose the authentication types that suit your data protection needs. Secure your system landscape by authenticating application users, establishing SSO connections to back-end systems, and protecting data and other resources.

Table 5: Security Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Secure back-end connections</td>
<td>• Basic - HTTP.</td>
</tr>
<tr>
<td></td>
<td>• SAML SSO - form based.</td>
</tr>
<tr>
<td></td>
<td>• X.509 - certificate based.</td>
</tr>
<tr>
<td></td>
<td>• OAuth - access token based.</td>
</tr>
<tr>
<td>Propagate user context via HTTP headers</td>
<td>Enables the back end to use the information in a REST API header to identify the user who sent the request.</td>
</tr>
<tr>
<td>System for Cross-Domain Identity Management (SCIM)</td>
<td>Delegate basic authentication requests to SCIM for either SAP Cloud Platform mobile service for development and operations applications or all requests from the SAP Cloud Platform tenant account.</td>
</tr>
<tr>
<td>Require two-factor authentication</td>
<td>Requires a one-time password for an application to implement two-factor authentication.</td>
</tr>
<tr>
<td>OAuth security configuration support</td>
<td>Authenticate hybrid applications using the OAuth protocol, which uses OAuth access tokens as credentials. Authenticate SAP Content to Go cards using the OAuth protocol. Third parties can access OAuth service provider resources without sharing passwords.</td>
</tr>
<tr>
<td>Authentication type OAuth-to-SAML Bearer Assertion</td>
<td>Enables client applications to use SAML assertions to access OAuth-protected resources.</td>
</tr>
<tr>
<td>Authenticating application users</td>
<td>Define the security configuration for an application to determine how application users are authenticated.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>SSO back-end connections</td>
<td>Assign an SSO mechanism to an application to define how the application can establish single sign-on connections with back-end systems. Features include:</td>
</tr>
<tr>
<td></td>
<td>- SAPAssertionSSO – select a PKCS #12 file to automatically populate the certificate and signing key properties.</td>
</tr>
<tr>
<td></td>
<td>- ApptoApp SSO mechanism – configure application-to-application communication, where the user is propagated with SAML 2.0 assertion generated from the destination.</td>
</tr>
<tr>
<td>Data protection and privacy</td>
<td>The mobile service does not store personal data. You can configure applications that you create to store sensitive data in an offline data store.</td>
</tr>
<tr>
<td>Support SAML assertion mapping</td>
<td>Administrators can define the roles that users must have to run an application. At runtime, application users are verified to have the required roles.</td>
</tr>
<tr>
<td>for applications</td>
<td></td>
</tr>
<tr>
<td>Virus scanning for back-end</td>
<td>Helps locate security issues and the communication time impacted.</td>
</tr>
<tr>
<td>connections</td>
<td></td>
</tr>
<tr>
<td>Unlock an application using a</td>
<td>Use a fingerprint, instead of a password, to unlock applications. Fingerprinting is one form of biometric authentication.</td>
</tr>
<tr>
<td>fingerprint as identification</td>
<td></td>
</tr>
<tr>
<td>Secure Login Server (SLS)</td>
<td>Select a secure login server profile for an application. This provides a channel for the device application to retrieve a client certificate from the secure login server.</td>
</tr>
</tbody>
</table>
1.1.2.6  App Management

Define, manage, and monitor your native, hybrid, Agentry, and Web apps through their entire life cycles.

Table 6: App Management Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manage and monitor applications</td>
<td>Use the Mobile Service for Development and Operations cockpit to manage and monitor device applications. The user interface is easy to understand and navigate. Change how applications are presented by toggling between list view and tile view. Icons and links enable you to easily access tools and information. Filters enable you to eliminate data and focus on what you want to see.</td>
</tr>
<tr>
<td>Manage and monitor Agentry Cloud Edition applications</td>
<td>Use the Mobile Service for Development and Operations cockpit to manage and monitor Agentry Cloud Edition applications. View information on account type, versions, and status that is retrieved from the Agentry component.</td>
</tr>
<tr>
<td>Define applications</td>
<td>Register mobile apps via the cockpit. Once defined, you can manage and monitor the apps.</td>
</tr>
<tr>
<td>Create back-end connections</td>
<td>Create secure back-end connections.</td>
</tr>
<tr>
<td>Ignore case for user names</td>
<td>By default, the platform uses case-sensitive matching to compare a registered user name for subsequent application access, and for push notification. This ensures that a connection is not shared by different users. Optionally you can override the default, and use case-insensitive matching for an application. This is useful for Fiori applications that expect user names to be matched case-insensitively, as in ABAP systems.</td>
</tr>
<tr>
<td>View application settings</td>
<td>View a summary of settings and URLs associated with a specific application. This enables you to see all the values in one place, and to copy and paste values as needed.</td>
</tr>
<tr>
<td>Import a Fiori application</td>
<td>You can import Fiori applications and connections that were developed in SAP Cloud Platform mobile service for app and device management, but you cannot edit their properties. This enables you to securely monitor the app via the cockpit.</td>
</tr>
<tr>
<td>Enable cross-origin policy</td>
<td>By default, the platform uses a same-origin policy. You can configure a cross-origin policy in the cockpit, which enables cross-origin resource access for applications.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>Configure cross-origin resource sharing (CORS)</td>
<td>Define how resources in one origin can access resources in another origin. Configure cross-origin resource sharing header parameters for the system using the cockpit. If header parameters are not configured, no cross-origin resource sharing is permitted.</td>
</tr>
<tr>
<td>Import applications from another origin (currently limited to SAP Cloud Platform mobile service for SAP Fiori)</td>
<td>The origin policy defined on the server determines the access rights for applications and connection objects imported from another origin. The origin policy cannot be modified by users. Fields and buttons may be dimmed in the cockpit, or not appear, according to the policy.</td>
</tr>
<tr>
<td>Enable cross-site request forgery (CSRF) protection</td>
<td>(Web applications only) When selected, all services, such as registration, are protected with CSRF tokens. Proxied endpoints are not protected, since they may be protected on the back end.</td>
</tr>
<tr>
<td>SAP Discovery Services</td>
<td>Publish app configurations to SAP Discovery Services, from which mobile apps can find their connection settings. Update or delete published discovery services.</td>
</tr>
<tr>
<td>Inactive applications</td>
<td>Configure how long an application registration can be inactive before it is deleted automatically. Once the period of inactivity expires, the platform automatically deletes the application registration. The application user must reconnect to start again.</td>
</tr>
<tr>
<td>Feature restriction policies</td>
<td>The administrator can manage a list of feature restriction policies for all applications from the cockpit. Each feature restriction policy works as a template. The administrator can add, delete, or update each template. An updated template is automatically applied to new hybrid applications, and can be manually applied to existing ones. The feature restriction policy can be overridden and customized at the application level.</td>
</tr>
<tr>
<td>Resource bundles</td>
<td>Resource bundles can be associated with client resources.</td>
</tr>
<tr>
<td>Custom Rewrite URL</td>
<td>For application request and response messages, you can set the Rewrite Mode to Custom Rewrite URL, and define both a search string and a replacement string. Use this feature to redirect URLs, to change an absolute URL to a relative URL, or to replace any string.</td>
</tr>
</tbody>
</table>
Feature | Description
--- | ---
Override the SCIM URL for an application | Define the URL that an application uses to delegate basic authentication requests to the System for Cross-Domain Identity Management (SCIM) server.

Manage registrations, users, and connections | Manage registrations, users, and connections from the cockpit.
Features include:
- Manage user consent for storing usage data. To ensure privacy, the user must give consent before usage logs can be stored. The consent can be revoked later. You can now view the date and time at which the consent was given/revoked.

View log and trace information | View log messages and trace information for a single application, or multiple applications. Search within the log and trace information to find specific items.

Maintenance mode | When runtime Java applications are under maintenance, relevant error messages are shown to the user.

### 1.1.2.7 App Updates

Intelligent update mechanisms keep your apps and app configurations up-to-date through optimized downloads, and allow you to maintain multiple app versions in the field.

**Note**
The App Update feature is only supported by applications that are created using the hybrid and App Modeler templates.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Version control for all application types</td>
<td>Administrators can activate or deactivate specific versions of an application. Users cannot run an inactive application version; they must upgrade their application to an active version.</td>
</tr>
</tbody>
</table>
### Feature

**AppUpdate version monitoring for hybrid apps per device**

(Hybrid only) View the distribution of application versions from the cockpit, for apps that are running on registered devices.

Features include:

- Use cockpit tools to sort table results, customize the table view, and export the table data from a specific application to a CSV file.
- To ensure privacy, the user must give consent before usage logs can be stored. The consent can be revoked later. View the date and time at which the consent was given/revoked.

**Optimize life-cycle management for hybrid applications**

To optimize life-cycle management and to provide more efficient client updates, a limited number of hybrid applications are maintained in the database.

**OData provisioning capability in SAP Cloud Integration (CI)**

OData provisioning capability supports REST, OData, and SOAP data sources. It converts the data sources to OData services and deploys them on HCI. The OData Services URL can be used to create back-end connections in the Mobile Service for Development and Operations cockpit.

*Note*

OData provisioning capability in SAP Cloud Integration and Back-end Connection Catalog Viewer are in Beta mode.

### 1.1.2.8 Usage Reporting

View app-specific usage statistics and reports.

#### Table 8: Usage Reporting Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Generate reports based on client usage data</td>
<td>Filter and view reports that are specific to client data by selecting the appropriate usage options. The usage options are based on user session, which is the duration for which the application is open.</td>
</tr>
<tr>
<td>Feature</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Audit reports include active user count</td>
<td>The active user count is included in an exported Global Licensing Audit Services (GLAS) report. You can also view the active user count in the cockpit.</td>
</tr>
<tr>
<td>Download CSV file of usage analytics</td>
<td>Download a CSV file containing usage analytics data, and import the file to Lumira, Excel, and so on.</td>
</tr>
</tbody>
</table>
| End-to-end HTTP traffic tracing in real time             | View end-to-end HTTP traffic between the application and the back end in real time for a single user/registration from the cockpit. You can also view log information for offline communication. This information is useful in an application development environment for testing, and in a production environment for troubleshooting. Features include:  
  - Outbound HTTP traffic - view outbound request traffic usage statistics for applications and connections. These values are for statistical comparison only, and are not comparable to values used for billing.  
  - Inbound HTTP traffic -  
    - Set thresholds for inbound registration, online, and offline requests. When a threshold is reached, requests are throttled and an HTTP error code is generated. The registration, online, and offline request thresholds apply to applications; the online request threshold applies to connections.  
    - View registration, online, and offline request hit counts (the number of times the threshold value is crossed) for applications and connections; and inbound request hit counts for connections.  
  - Trace network activity - generate and download network tracing reports, based on user name, connection, application, or content type. |
| Push notification statistics                             | View push notification statistics for push enabled apps as part of usage reporting. You can view notification details, notifications by applications, and notifications by operating system.                           |
1.1.2.9  SAP Content to Go

Share data from virtually any back-end to iPhones in a mobile-friendly format, either online or offline. When online, dynamic updates provide the most up-to-date business data available.

Table 9: SAP Content to Go Features

<table>
<thead>
<tr>
<th>Feature (by role)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Administrators</strong></td>
<td></td>
</tr>
<tr>
<td>Enable data mobilization from SAP Fiori desktop applications</td>
<td>Administrators define templates for the different types of Fiori application content that can be mobilized through the cockpit.</td>
</tr>
<tr>
<td><strong>End Users</strong></td>
<td></td>
</tr>
<tr>
<td>Mobilize data from SAP Fiori desktop applications</td>
<td>End users install the SAP Content to Go on their iPhones, then share content from Fiori applications via the SAP Fiori Launchpad on their desktop computers. While online, the SAP Content to Go mobile service provides dynamic updates. While offline, end users can still view content. Rapid card templates simplify card creation.</td>
</tr>
</tbody>
</table>

1.1.2.10  Build or Extend

Extend existing on-premise solutions, or use SAP Cloud Platform to build a fully cloud-hosted solution that uses HANA as the back end, the Java Runtime, HTML5 app, and mobile app tools.

Table 10: Cloud-Hosted Solution Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build a fully cloud-hosted solution</td>
<td>Use SAP Cloud Platform to build a fully cloud-hosted solution that uses a HANA back end, the Java runtime, an HTML5 app, and mobile app tools.</td>
</tr>
<tr>
<td>Extend an existing on-premise system into the cloud</td>
<td>Extend your existing on-premise system, either as a migration path toward a cloud-hosted solution, or a hybrid solution that mixes cloud and on-premise features.</td>
</tr>
<tr>
<td>Multiple landscapes</td>
<td>Manage both production and non-production (such as development and quality assurance) landscapes.</td>
</tr>
</tbody>
</table>
Flag dedicated account types for different uses in the landscapes

SAP Cloud Platform mobile service for development and operations provides dedicated accounts to use in either non-production or production landscapes. You can flag these account types for use in the landscape.

You can move account types, such as from development to quality assurance (both non-production landscapes), or from a non-production landscape to production landscape.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type of Change</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag dedicated account types for different uses in the landscapes</td>
<td></td>
<td>SAP Cloud Platform mobile service for development and operations provides dedicated accounts to use in either non-production or production landscapes. You can flag these account types for use in the landscape. You can move account types, such as from development to quality assurance (both non-production landscapes), or from a non-production landscape to production landscape.</td>
</tr>
</tbody>
</table>

## 1.2 What's New

New features and feature enhancements in SAP Cloud Platform mobile service for development and operations.

Table 11: New Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Type of Change</th>
<th>Description</th>
</tr>
</thead>
</table>
| Signing Profiles | New | You can now manage your enterprise’s signing profiles, which are used by the Cloud Build Service to build the Fiori Client IPA and APK files. See:  
  - Manage Signing Profiles  
  - Creating a Signing Profile  
  - Editing a Signing Profile  
  - Deleting a Signing Profile |
| Configure Fiori Client using its destination URL, and publish to the Discovery Service | New | You can configure an App Store Fiori Client from its destination URL, and publish the information to the Discovery Service from Mobile Service for Development and Operations cockpit. See:  
  - Defining Connectivity  
  - Managing Destinations  
  - Creating a Destination |
<table>
<thead>
<tr>
<th>Feature</th>
<th>Type of Change</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP Content to Go templates replaced</td>
<td>New</td>
<td>The Fiori Generic and Generic templates have been replaced with new templates: Sample Template Product and Sample Template Sales Orders. See: Creating Card Types, step 3</td>
</tr>
<tr>
<td>View Agentry server URL in configuration settings</td>
<td>New</td>
<td>You can view the Agentry server URL in the Configuration section of the Agentry Specific Settings tab. The Transmit URL replaces the Path URL used in previous releases. See: Configuring Agentry Settings, the Transmit URL setting under Configuration</td>
</tr>
<tr>
<td>Export the User Audit log</td>
<td>New</td>
<td>Export the User Audit log as a CSV file. See: Viewing the User Audit Log</td>
</tr>
<tr>
<td>OData Sample includes sample images</td>
<td>New</td>
<td>The OData Sample includes sample images with links to the images. The images are designed to be used with tutorials; you cannot upload new images. See: ● Using the Sample Back End ● Viewing Sample Pictures</td>
</tr>
<tr>
<td>SAP Cloud Platform mobile service for development and operations is now available in Korean (ko-KO).</td>
<td>New</td>
<td></td>
</tr>
<tr>
<td>Customize SAP Content To Go cards using JavaScript</td>
<td>New</td>
<td>You can now add JavaScript code in the HTML editor to customize Content To Go cards. Use D3.js to incorporate graphics in your cards. D3.js is a JavaScript library for producing dynamic, interactive data visualizations in web browsers. It makes use of the SVG, HTML5, and CSS standards. See: Creating Card Types, step 9</td>
</tr>
<tr>
<td>Add action parameters for SAP Content to Go cards</td>
<td>New</td>
<td>You can now add action parameters, which allow users to input parameter values while executing an action. See: Defining Actions</td>
</tr>
<tr>
<td>Feature</td>
<td>Type of Change</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>----------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Admin-defined notification message used in APNs for SAP Content to Go cards</td>
<td>New</td>
<td>Admins can now create the notification message that is sent when a card is created or updated. See: Creating Card Types</td>
</tr>
<tr>
<td>Cards with collection only OData service</td>
<td>New</td>
<td>Expect Collection allows you to get a response as JSON data or OData service via a collection-based query rather than an individual query request. See: Creating Card Types</td>
</tr>
</tbody>
</table>

You can view our release videos here: [https://www.youtube.com/playlist?list=PLWV533hWWvDkaoutrpnPj4ctLNX97vb](https://www.youtube.com/playlist?list=PLWV533hWWvDkaoutrpnPj4ctLNX97vb).

### Related Information

- Manage Signing Profiles [page 172]
- Defining Connectivity [page 58]
- Managing Destinations [page 134]
- Creating a Destination [page 135]
- Creating Card Types [page 145]
- Configuring Agentry Settings [page 108]
- Viewing the User Audit Log [page 171]
- Using the Sample Back End [page 86]
- Viewing Sample Pictures [page 89]
- Creating a Signing Profile [page 172]
- Editing a Signing Profile [page 173]
- Deleting a Signing Profile [page 174]
- Defining Actions [page 151]

### 1.3 Get Started

SAP Cloud Platform mobile service for development and operations offers authentication, secure on-boarding, native push notifications, and reporting capabilities for enterprise mobile applications. Mobile Service for Development and Operations cockpit provides a single comprehensive Web administration and monitoring portal for configuring and managing mobile applications.

In Mobile Service for Development and Operations cockpit, set up developer and customer accounts.

Enable Developer Accounts [page 22]
A developer account allows you to explore the basic functionality of SAP Cloud Platform mobile service for development and operations. Access is open to everyone. A developer account is also known as a trial account.

Set Up Customer Accounts [page 23]
An SAP Cloud Platform mobile service for development and operations customer account allows you to host production, business-critical applications that are supported 24/7. A customer account is also called a production account.

Define Customer Account Types [page 26]
You can define a customer account type as either production or nonproduction. Use a production account in a production system. Use a nonproduction account for development and quality-assurance testing.

Set Up Customer Accounts for Agentry [page 27]
An account administrator can set up Agentry landscapes for customers to develop, test and run Agentry mobile applications. To do this an account should be set up separately to consume each service such as development, preview, test and production usage.

1.3.1 Enable Developer Accounts

A developer account allows you to explore the basic functionality of SAP Cloud Platform mobile service for development and operations. Access is open to everyone. A developer account is also known as a trial account.

Context

For information about how to register and create a developer account, see Signing Up for a Developer Account.

Procedure

1. Log in to SAP Cloud Platform Cockpit, and select Services.
2. Under Mobile Services, select Development & Operations.
3. Click Enable.
1.3.2 Set Up Customer Accounts

An SAP Cloud Platform mobile service for development and operations customer account allows you to host production, business-critical applications that are supported 24/7. A customer account is also called a production account.

Prerequisites

To explore and use the powerful capabilities of Mobile Service for Development and Operations cockpit, verify that:

- You have purchased a license for SAP Cloud Platform mobile service for development and operations and received an e-mail notification that the service has been configured for your account.
- SAP has enabled your account for SAP Cloud Platform mobile service for development and operations by:
  - Subscribing your account to the mobile HTML5 application, which is available under the hanamobileprod Provider account.
  - Subscribing your account to the mobilejava JAVA application, which is available under the hanamobileprod Provider account.
  - Enabling the Principal Propagation property under Trust > Local Service Provider.
- A single user within your organization has HanaMobileAdmin permissions, and is known as a super administrator; this user has full access to SAP Cloud Platform Cockpit and can assign subscriptions for Mobile Service for Development and Operations cockpit to your organization’s SAP Cloud Platform account.

Note

Do not change any of these settings.

Context

To set up the Mobile Service for Development and Operations cockpit and integrate it into your internal landscape, a super administrator must perform the following steps.

Procedure

1. Log in to SAP Cloud Platform Cockpit, and in the left pane, select Services.
2. Under Mobile Services, verify that Development & Operations is enabled for your account.
3. (Recommended) Create a user group and assign users to it to access the Mobile Service for Development and Operations cockpit:
   a. In the left pane, select Authorizations.
c. Click New Group, enter a group name, and click Save.
d. To the right of Individual Users, click Assign, and enter a user to assign to the group.

4. In the left pane, select Services, then select Development & Operations:

5. Assign the required privileges to the users who will access the Mobile Service for Development and Operations cockpit:
   ○ For a Java application subscription:
     2. Select Administrator, and assign either an individual user or a group of users to it.
     The table below describes the predefined roles you can assign to a group of users:

     | Role            | Purpose                                                                                                                                 |
     |-----------------|------------------------------------------------------------------------------------------------------------------------------------------|
     | Helpdesk        | Read-only privileges for viewing administrative operations in Mobile Service for Development and Operations cockpit. Helpdesk operators review system information and determine the root cause of reported problems. |
     | Administrator    | To create, read, update, and delete any object in Mobile Service for Development and Operations cockpit.                                    |
     | Notification User| To send push notifications to applications that use REST services, and to access Java application subscriptions. Also used to send push notifications to all users of a specific application using the Push Desk. |
     | Developer        | (For Kapsel application development) To upload, delete (self-configured), and query Kapsel applications. Does not include permission to change the status of Kapsel applications, only an administrator is permitted to change the status. Also enables application users to access the sample OData service from an application, which the Administrator has defined and configured access to the sample OData service. |

   ○ For an HTML5 application subscription:
     By default, the HanaMobileAdmin permissions are granted to the AccountAdministrator role.

   **Note**
   An account administrator has permission to access Mobile Service for Development and Operations cockpit via SAP Cloud Platform Cockpit. To give other users in your organization this permission, follow step 3, and assign the AccountAdministrator role to the group; or you can assign the AccountAdministrator role to individual users.

   To add an administrator for Mobile Service for Development and Operations cockpit only:
   2. (Optional) To create a new role, select Roles New Role, and enter a role name, for example, MobileServicesCockpitAdministrator. Save your changes.
   3. (Optional) To assign a user to the new role, click Assign, enter the user ID, and click Assign.
   4. In the left pane, select Destinations & Permissions.
5. Under Application Permissions, select HanaMobileAdmin, and click Edit.

6. Expand the Assigned Role list, select the role you created (in this example, MobileServicesCockpitAdministrator), and click Save.

   The HanaMobileAdmin permissions are granted to the MobileServicesCockpitAdministrator role, and revoked from the AccountAdministrator role.

   **Note**
   The HanaMobileAdmin permissions can be granted to only one role at a time.

7. Grant the new role to users who need administrative access to Mobile Service for Development and Operations cockpit, either:
   - Individual users, or
   - A group, and then add users to the group.

6. For application connections such as onboarding, use one of the following base URLs:

7. To access Mobile Service for Development and Operations cockpit, use one of the following URLs:
   - For accounts on the ap1.hana.ondemand.com host, use .

   See the complete list of SAP UI5 supported browsers on SAP Help Portal.

   If a Mobile Service for Development and Operations cockpit session times out and displays a blank screen, refresh or restart the browser.

8. (Optional) To access on-premise back ends, install and configure SAP cloud connector. See Installing the Cloud Connector.

   **Note**
   In Mobile Service for Development and Operations cockpit, configure on-premise HTTPS back-end connections as HTTP using the virtual host address. Communication between the cloud and your on-premise SAP cloud connector is secure. Communication between SAP cloud connector and your back-end system uses standard HTTPS security.
a. In SAP cloud connector, verify that necessary back-end service URLs are whitelisted. Every on-premise URL that is configured in Mobile Service for Development and Operations cockpit, such as application endpoints or the security configuration, must be whitelisted. See Configuring Access Control (HTTP).

b. Generate a system certificate and import it into SAP cloud connector. See Installation of a System Certificate for Mutual Authentication.

9. Access one of the Mobile Service for Development and Operations cockpit URLs described in step 7.


11. Use the REST client to test the application configuration. See REST API Application Development Overview.

Related Information

Troubleshooting: Common Issues [page 221]
Configuring Applications [page 43]
Native Push Notification for a Back End [page 245]
REST API Application Development Overview [page 224]

1.3.3 Define Customer Account Types

You can define a customer account type as either production or nonproduction. Use a production account in a production system. Use a nonproduction account for development and quality-assurance testing.

Context

Production and non-production accounts help customers to distinguish between their environments and also enable them to know what their landscape is used for. There aren’t any functional differences between them.

Note

More resources are provisioned to productive landscape. After you define an account as a production type, you cannot change it to a nonproduction type. For more information on production and nonproduction accounts, see SAP Cloud Platform Mobile Service for Development and Operations [page 3].

Procedure

1. In Mobile Service for Development and Operations cockpit, select ➤ Settings ➤ Account Type ➤
2. Under **Change Account Type**, define the current account type, and save your settings:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Name</td>
<td>Name of the account.</td>
</tr>
<tr>
<td>Display Name</td>
<td>The display name of the account.</td>
</tr>
<tr>
<td>Description</td>
<td>(Appears only for a nonproduction account type) Enter a description, such as development or test.</td>
</tr>
<tr>
<td>URL</td>
<td>URL to the landscape with which the account is associated.</td>
</tr>
<tr>
<td>Productivity System</td>
<td>Select whether the account type is production or nonproduction.</td>
</tr>
</tbody>
</table>

**Results**

Under **Account List**, you see the accounts that are currently defined.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Account Name</td>
<td>The name of the account.</td>
</tr>
<tr>
<td>Display Name</td>
<td>The display name of the account.</td>
</tr>
<tr>
<td>URL</td>
<td>URL to the landscape with which the account is associated.</td>
</tr>
<tr>
<td>Production</td>
<td>Indicates whether the account type is production or nonproduction.</td>
</tr>
</tbody>
</table>

**1.3.4 Set Up Customer Accounts for Agentry**

An account administrator can set up Agentry landscapes for customers to develop, test and run Agentry mobile applications. To do this an account should be set up separately to consume each service such as development, preview, test and production usage.

**Context**

The administrator can create Agentry Development account, Preview account, Test account, or Production account.

- To create an Agentry account, see [Creating Accounts](#).
- To create a new account for Agentry development, ensure you allocate a JAVA quota, see [Managing Account Quota](#).
You can enable the Agentry account by following the steps below:

**Procedure**

1. In SAP Cloud Platform Cockpit, select a global account.
2. Select the respective Agentry account and then, Services on the left pane.
3. In the Agentry section, select Agentry > Enable.

**Note**

Also enable Development & Operations Preview if the Agentry account is used for development or preview. Enable Development & Operations if the Agentry account is for test or production usage. At a time, only four accounts can be registered for a customer.

- To understand how to use a service, see Using Services
- To setup cloud connector for the on premise backend, see SAP Cloud Platform Connector
- If your account hasn’t been provisioned, see Provisioning an Agentry Account [page 153]

### 1.4 Administration

Administrators interact with SAP Cloud Platform mobile service for development and operations to ensure the production environment works efficiently.

Administrator tasks fall into two main categories:

- Application administration for configuring applications for deploying to users and monitoring application in the user community.
- Security administration for determining the Cloud Platform security features used for mobile applications.

**SAP Cloud Platform Mobile Service for Development and Operations Overview [page 30]**

SAP Cloud Platform mobile service for development and operations provides services to mobile applications, such as application analytics, app resources, onboarding, HTTP/HTTPS configuration and so on.

**User Interface [page 41]**

Frequently used icons in Mobile Service for Development and Operations cockpit. Actual icon styles may vary slightly from what is shown.

**Application Administration [page 42]**

Use Mobile Service for Development and Operations cockpit and other tools to manage and monitor native, hybrid, and Web mobile applications. Managing includes defining and configuring applications; monitoring applications and application usage; viewing statistics and logs; checking system health; and troubleshooting problems.

**System Administration [page 177]**

At the system level, you can configure Cross-Origin Resource Sharing and the System for Cross-Domain Identity Management authentication.
Security Administration [page 182]
An essential element when planning your mobile service for development and operations landscape is to provide secure propagation of mobile users' identities to back-end systems. The product supports a range of popular application authentication protocols and maps them to back-end systems.

Maintenance Mode [page 221]
When runtime Java applications are in maintenance mode and if you log in to the Mobile Service for Development and Operations cockpit, there are different messages you may see.

Troubleshooting: Common Issues [page 221]
Overview of common issues.

Related Information

Application Administration [page 42]
Security Administration [page 182]
1.4.1 SAP Cloud Platform Mobile Service for Development and Operations Overview

SAP Cloud Platform mobile service for development and operations provides services to mobile applications, such as application analytics, app resources, onboarding, HTTP/HTTPS configuration and so on.

Mobile application services consist of the following:

- **Application Analytics** – usage statistics that can be displayed graphically in Mobile Service for Development and Operations cockpit.
- **App Resources** – containers of dynamic configurations, styles, or content that can be downloaded by native applications.
- **Onboarding** – authentication of users who are registering through SAP Mobile Place.
- **HTTP/HTTPS Configuration** – open standards for client communications.
- **Lifecycle Management** – managing and deploying multiple versions of an application.
- **Offline OData Service** – optimizes data transport between the back end and the client offline store.
- **Push Notifications** – native notifications sent from back-end systems to the server, which forwards them on to the clients.
• Mobile Service for Development and Operations cockpit – deploying, managing, and monitoring applications.
• Supportability – logs for monitoring system health and troubleshooting.

SAP Cloud Platform mobile service for development and operations can expose on-premise back-end services through SAP Cloud Connector, and on-demand back-end services directly.

Cloud Platform security enables you to use an on-premise identity management system for on-demand applications. You can use basic authentication using LDAP, or form-based application authentication using SAML.

All configuration and runtime data is persisted in an SAP HANA database.

Logging and Tracing Overview [page 32]
SAP Cloud Platform mobile service for development and operations provides supportability through logs and traces that enable administrators, developers, and support professionals to troubleshoot application issues. All logs use a common format. Information and Warnings and Errors logs are stored in the server database. All log entries for a particular business or application flow (such as an OData request or a registration) are correlated across the client and server stack, providing an end-to-end flow, which helps identify the source of a problem.

Application Usage Reporting Overview [page 32]
You can collect standard usage information for applications, and view reports based on information logged by clients and uploaded to the server.

Offline Applications Overview [page 33]
Offline support enables client applications to access back-end data without establishing a connection to the back end.

Discovery Service Overview [page 35]
The SAP Discovery Service provides the configuration information necessary for a user without enrolling a device with SAP Cloud Platform mobile service for app and device management. This service enhances the user onboarding process by letting you distribute initial configuration data to mobile apps.

Push Overview [page 35]
Use the push feature to push updates from the back-end data source to applications that are running on mobile devices. The back end can also push notifications to apps that provide a certain capability or run on a specific device type, rather than to particular applications.

Hybrid App Feature Restriction Overview [page 38]
You can set feature restriction policies for hybrid apps in Mobile Service for Development and Operations cockpit. This gives you additional control over the SAP Fiori Client application features that are allowed and restricted.

Client Database Upload Overview [page 38]
End users can securely upload local database files from a supported device to the server for analysis.

Origin Policy Overview [page 39]
SAP Cloud Platform mobile service for development and operations supports applications from another origin, currently SAP Cloud Platform mobile service for SAP Fiori.

SAP Content to Go Overview [page 40]
(iPhone only) SAP Content to Go enables end users to mobilize Fiori app web content without installing a custom mobile application. Administrators can manage SAP Content to Go service from the cockpit.
1.4.1.1 Logging and Tracing Overview

SAP Cloud Platform mobile service for development and operations provides supportability through logs and traces that enable administrators, developers, and support professionals to troubleshoot application issues. All logs use a common format. Information and Warnings and Errors logs are stored in the server database. All log entries for a particular business or application flow (such as an OData request or a registration) are correlated across the client and server stack, providing an end-to-end flow, which helps identify the source of a problem.

System logs collect log messages that allow administrators and support professionals to identify problem areas. Developers can identify code problems by capturing debug-level log messages. You can control the amount of information that is captured by setting the log level for individual component. Application tracing captures additional business data for a request (such as message data, HTTP headers, and URIs), which you can use to troubleshoot application problems. Enable tracing for individual logging components on an as-needed basis. Logs and traces collected on a device are automatically synchronized with the server when an application is launched or switched to foreground or background on the client.

Related Information

- Setting Log Levels [page 162]
- Enabling Application Traces [page 163]
- Viewing Event Logs [page 165]

1.4.1.2 Application Usage Reporting Overview

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

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

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

The administrator has complete control over the usage reports upload in the SAP Cloud Platform mobile service for development and operations, and can view reports and carry out necessary operations.

Enable Usage Collection for an Application

The administrator can configure the uploaded records in the server remotely using Mobile Service for Development and Operations cockpit and optimize them using WiFi. This process minimizes the impact of usage collection on the end user’s cellular data plan.
Enable Application-Specific Columns in the Database

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

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

Set the Maximum Threshold for Storing Records

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

Related Information

- Defining Usage Report Policy [page 55]
- Viewing Server Data Report [page 159]

1.4.1.3 Offline Applications Overview

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

You might want to run applications offline to:

- Improve performance by accessing offline data instead of sending data requests to SAP Cloud Platform mobile service for development and operations.
- Enable users to continue to use applications when there is intermittent network coverage.
- Support business processes that must be executed by a user while the application is offline.

To work offline, an application must initialize an offline store, which stores data that the application can access when it is offline. SAP Cloud Platform mobile service for development and operations provides an Offline OData Service that moves data between the back end and the client offline store.
SAP Cloud Platform mobile service for development and operations retrieves data from an OData producer that is running in a back end, and from that data creates an initial database on the client. On an ongoing basis, SAP Cloud Platform mobile service for development and operations updates the client database based on changes, or deltas, that have been made to the data on the back end. Deltas between the back-end data and the client data are identified either by the back end or by SAP Cloud Platform mobile service for development and operations.

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

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

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

**Related Information**

*Defining Offline Settings for Applications [page 70]*
1.4.1.4 Discovery Service Overview

The SAP Discovery Service provides the configuration information necessary for a user without enrolling a device with SAP Cloud Platform mobile service for app and device management. This service enhances the user onboarding process by letting you distribute initial configuration data to mobile apps.

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

- Native OData App Development. Navigate to XXX Applications > Developing with MAF Logon for XXX > Onboarding with SAP Mobile Place, where XXX is the platform name: iOS, Android, or Windows.
- Certificate Delivery Services Provider Plugin.
- Provisioning Applications Using Mobile Place
- App Configuration Using a Discovery Service Provider.

After the Discovery Service procedure is added to the application, use the cockpit for the mobile service for app and device management to publish application configurations to the SAP Discovery Service, on which mobile applications can find their connection settings. You can publish the default application configuration, or customize it by adding either name:value pairs or a JSON construct.

Related Information

Enabling Applications to Discover Configurations [page 97]

1.4.1.5 Push Overview

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

You can use SAP Cloud Platform mobile service for development and operations to manage push for individual applications that use native notifications.

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

**Push Notification**

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

**Push Desk Notification**

For native and hybrid applications, Administrators can send push notifications to all users of a push-enabled application. This is useful to broadcast important information, such as a maintenance window or a new feature or promotion. The Push Desk notification tool enables you to create a filtered list of recipients and compose the message. Requires the Notification User role to send notifications. Without the role you can still access the Push Desk tool and filter recipients, but you cannot send notifications.

**Custom Push Provider**

The custom push provider enables you to configure applications to receive push notifications via a private push vendor. For example, for an application running on Android, you may want to replace the GCM push service for a local vendor service. Configure the custom push provider for specific device types.

**Predefined Push Configuration**

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

**Push Statistics**

View push notification statistics for push-enabled apps from *Push Statistics*. You can see notification details, notifications by applications, and notifications by operating system.
Capabilities-based Push Support

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

Device-type (form factor) Support

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

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

Configuration of Capabilities

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

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

Actionable Push

Starting with iOS 8, Apple supports actionable push notifications. The push API offered with SAP Cloud Platform mobile service for development and operations has been enhanced to support this feature. Using the API, the back end provides the capability for back ends to send the push ‘category’ through the platform to the device. This change applies to iOS only; for Android actionable push is fully controlled on the device by the app.

- **Note**
  Hybrid apps do not support actionable Push.
Related Information

REST API Application Development Overview [page 224]

1.4.1.6 Hybrid App Feature Restriction Overview

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

When you edit a hybrid app in Mobile Service for Development and Operations cockpit, available feature plugins are listed on the Client Policies screen. You can indicate features that should be restricted from the user. Feature plugins are typically JavaScript APIs that provide access to the native APIs of the mobile device (implemented as Apache Cordova plugins). Plugins include:

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

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

1.4.1.7 Client Database Upload Overview

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

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

Developers (Application Development)

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

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

Device Users

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

Developers (Troubleshoot)

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

1.4.1.8 Origin Policy Overview

SAP Cloud Platform mobile service for development and operations supports applications from another origin, currently SAP Cloud Platform mobile service for SAP Fiori.

Origin Policy

The origin policy determines the access rights for application and connection objects coming from another origin. The origin policy is defined on the server, and cannot be modified by users.

Support for Origin Policies

Mobile Service for Development and Operations cockpit supports these policies for applications and connections. For example, an application or connection may appear in Mobile Service for Development and Operations cockpit with fields and buttons that are grayed out, or do not appear, according to the policies.
1.4.1.9 SAP Content to Go Overview

(IPhone only) SAP Content to Go enables end users to mobilize Fiori app web content without installing a custom mobile application. Administrators can manage SAP Content to Go service from the cockpit.

SAP Content to Go Users

End users can download the SAP Content to Go application, which allows them to display business content in the form of cards. They can subscribe to the cards from the app itself, and subscribed card instances are delivered to their devices. End users cannot delete these cards, but they can unsubscribe the card type. App users can publish content from Fiori systems that are configured to support Content to Go. To use Fiori smart templates, select suitable objects or overview pages, and click Add to Mobile.

Administrators

You can use the cockpit to define and configure the SAP Content to Go application as an SAP Card application type. You can manage the SAP Content to Go service using Mobile Application SAP Content to Go.

Note

For information about how to register Fiori Smart Template Cards from the Fiori launchpad to the mobile service for development and operations Content to Go service, refer to SAP Note 2424974.

Services

The mobile service for development and operations manages the card decks. APNS pushes updated content to the Content to Go app.

Related Information

Configuring SAP Content to Go Applications [page 95]
1.4.2 User Interface

Frequently used icons in Mobile Service for Development and Operations cockpit. Actual icon styles may vary slightly from what is shown.

The cockpit has an enhanced user interface which is now the default view. When you log on, you are prompted to choose whether you want to access the enhanced user interface or the older one. The old user interface does not have the new features, hence, this is not recommended.

Table 14: Mobile Service for Development and Operations cockpit Icons

<table>
<thead>
<tr>
<th>Icon</th>
<th>Purpose</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>📅</td>
<td>Toggle Menu</td>
<td>Toggle between showing and hiding the navigation menu in the left pane. Display the menu for fast, easy navigation; hide it to maximize the window area.</td>
</tr>
</tbody>
</table>
| 🏡 | Home | Go to the Mobile Service for Development and Operations cockpit home screen. Links to frequently used tasks, and useful information are provided, and current event log XML files are available. The title indicates “Preview” if you are accessing the preview system, to distinguish it from the production system.  

Note: You may need to configure Internet Explorer 11 to view the XML event log files:  
1. Go to Internet Options ➤ Content ➤ Feeds and Web Slices ➤ Settings ➤ 
2. Uncheck “Turn on feed reading view”. |
| 📅 | About | View information about the platform. |
| 📜 | Help | Access online documentation on the help portal. |
| 🙌 | My Profile | View information about the current user profile and customer account. |
| 🕵️‍♂️ | Log Out | Log out of the application. |
| 📖 | Create | Configure a new resource for a feature, such as a configuring a new destination from within Connectivity. |
| 🛠️ | Action Settings | View an available action, such as customizing table columns. |
| 🔔 | New | Add a new item, for example, a destination, a provider, or a feature restriction policy. |
| 🕺 | Ping | Ping a destination. |
1.4.3 Application Administration

Use Mobile Service for Development and Operations cockpit and other tools to manage and monitor native, hybrid, and Web mobile applications. Managing includes defining and configuring applications; monitoring applications and application usage; viewing statistics and logs; checking system health; and troubleshooting problems.

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

The administrator creates an application definition in Mobile Service for Development and Operations cockpit, which includes the unique application identifier, plus the connection to its back-end data source in the production system, the security configuration, and application-specific entries.

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

Configuring Applications [page 43]
Create an application definition that enables you to manage the application using Mobile Service for Development and Operations cockpit. The application definition includes a unique application identifier, connections to the back-end data source, and optionally, other feature settings.

Configuring SAP Content to Go Applications [page 95]
You can configure the predefined SAP Content to Go application from the cockpit for the development and operations mobile service.

Enabling Applications to Discover Configurations [page 97]
Using Mobile Service for Development and Operations cockpit, you can publish application configurations to the SAP Discovery Service, on which mobile applications can find their connection settings. You can update or delete published configurations at any time.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Purpose</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>🌐</td>
<td>OData Application Destination Test</td>
<td>Test OData application destination.</td>
</tr>
<tr>
<td>🔢</td>
<td>Sort</td>
<td>Sort a list based on criteria you choose, such as ascending or descending order, or the column name.</td>
</tr>
<tr>
<td>🔍</td>
<td>Details</td>
<td>Open a dialog to see details of the selected resource.</td>
</tr>
<tr>
<td>⚒️</td>
<td>User Settings</td>
<td>Opens a dialog box to choose whether you want to use the enhanced user interface or the old version.</td>
</tr>
</tbody>
</table>
Configuring Agentry Applications [page 102]
Create an application definition that enables you to manage an Agentry application using the cockpit. The application definition includes Name, Application ID, Vendor and the Creation Date.

Defining Agentry Application Settings [page 106]
(Optional, applies only to Agentry) Configure application-specific settings for the selected application, using mobile service for development and operations Cockpit, configuration files, or other tools.

Managing and Monitoring Applications [page 120]
Use Mobile Service for Development and Operations cockpit to manage applications, registrations, users, back-end connections to the data source; view application usage statistics; and manage and view application reports.

1.4.3.1 Configuring Applications

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

To configure an application, you must provide a destination. Other settings are optional.

Defining Applications [page 43]
Create a new application definition, which enables you to use Mobile Service for Development and Operations cockpit to manage the application.

Managing Application Features [page 45]
You can add features to an application from the Features menu option.

Configuring Assigned Features [page 46]
Configure features associated with an application definition.

Related Information

Set Up Customer Accounts [page 23]
Migrating to SAP Cloud Platform Mobile Service for Development and Operations [page 412]

1.4.3.1.1 Defining Applications

Create a new application definition, which enables you to use Mobile Service for Development and Operations cockpit to manage the application.

Context

When you define an application, you can choose a template, or you can just create a shell:
If you choose one of the templates for a native, hybrid, or Web application, available features for the template are included, including security. Later you can add features that make sense for the application.

If you choose to create a shell, the security feature is added automatically, and you can add any other features you want.

In a separate step, configure the assigned features, or add more features. See Managing Application Features and Configuring Assigned Features.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select [Mobile Applications Native/ Hybrid] and click New.
2. In New Application, enter:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config Templates</td>
<td>Leave blank if you want to define a shell application, and add features later. Otherwise select a configuration template:</td>
</tr>
<tr>
<td></td>
<td>○ Native – native applications, including Android, BlackBerry, iOS, Windows Mobile 8, and Windows 8.</td>
</tr>
<tr>
<td></td>
<td>○ Hybrid – Kapsel container-based applications.</td>
</tr>
<tr>
<td></td>
<td>○ Web – applications that run in a Web browser.</td>
</tr>
<tr>
<td></td>
<td>○ Fiori – Kapsel container-based applications that include the Cloud Build feature.</td>
</tr>
<tr>
<td></td>
<td>○ App Modeler - metadata-based applications.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique identifier for the application, in reverse-domain notation. This is the application or bundled identifier that is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Cloud Platform mobile service for development and operations, and client applications use the Application ID when sending requests to the server. An application ID:</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>○ Can contain up to 64 characters</td>
</tr>
<tr>
<td></td>
<td>○ Cannot include spaces</td>
</tr>
<tr>
<td></td>
<td>○ Cannot begin with a period, and cannot contain two consecutive periods</td>
</tr>
<tr>
<td></td>
<td>○ Cannot be any of these case-sensitive keywords: Admin, AdminData, Push, smp_cloud, resource, test-resources, resources, Scheduler, odata, applications, Connections, public, lcm</td>
</tr>
<tr>
<td></td>
<td>We recommend that you assign IDs that contain a minimum of two periods, for example, com.sap.mobile.appl.</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>----------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Name</td>
<td>Application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) The description can contain up to 255 alphanumeric and special characters.</td>
</tr>
<tr>
<td>Vendor</td>
<td>(Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long.</td>
</tr>
</tbody>
</table>

1.  
2.  
3. Click Save. The Info page appears with current settings.

**Related Information**

Application Authentication [page 189]
Defining Access Control [page 48]
Configuring SCIM Authentication [page 179]
User Identity Propagation Methods [page 183]

### 1.4.3.1.2 Managing Application Features

You can add features to an application from the Features menu option.

**Context**

Features enable capabilities that developers have added to an application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Features**.
2. (Optional) To find out more information about a feature, click the feature name.
   - Under **Feature Details**, read the feature summary.
   - Click **Show APIs** to access the SAP API Business Hub and view API information for SAP Mobile Services (Push Notification only).
3. Click **Add to Application**.
### Table 16: Basic Features

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access Control</td>
<td>Enable role checking and application version checking.</td>
</tr>
<tr>
<td>App Update</td>
<td>Upload new versions of a hybrid application.</td>
</tr>
<tr>
<td>Client Policies</td>
<td>Configure native client policies for an application.</td>
</tr>
<tr>
<td>Client Resources</td>
<td>Add client resources to an application.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Configure routes to additional back ends.</td>
</tr>
<tr>
<td>Document Repository</td>
<td>Enable access to the document repository without registering.</td>
</tr>
<tr>
<td>Offline</td>
<td>Enable secure, offline access to data on the device.</td>
</tr>
<tr>
<td>Push Notification</td>
<td>Register devices to receive native push notifications.</td>
</tr>
<tr>
<td>Sample Back End</td>
<td>Use the sample service during development and testing.</td>
</tr>
<tr>
<td>Security</td>
<td>Implement security related SDK services to authenticate users.</td>
</tr>
<tr>
<td>Cloud Build</td>
<td>Build Custom Fiori Clients for your Fiori Launchpad or individual SAP Fiori business apps, and enable them to use the SAP Mobile Platform SDK.</td>
</tr>
<tr>
<td>Agentry</td>
<td>Configure a new Agentry cloud application.</td>
</tr>
</tbody>
</table>

4. In **Select Application**, select an application from the list. You can filter the list of options to shorten it. If you don’t see the application you want, you can click **New** to define one, as described in *Defining Applications*.

5. Click **OK**. The new feature appears in the application, and you can configure its settings as described in *Configuring Assigned Features*.

### 1.4.3.1.3 Configuring Assigned Features

Configure features associated with an application definition.

### Context

You can add additional features, and delete features that are not required. Some features require coding in the application. In general, there should be no dependency between kits.
Procedure

1. In Mobile Service for Development and Operations cockpit, select \( \text{Mobile Applications} \ \text{Native/Hybrid} \) and select an application.

2. Under \textit{Assigned Features}, click the Add icon \( + \), select a feature, and click \textit{OK}.

3. Configure the feature for the selected application, and \textit{Save}.

- \textbf{Defining Access Control [page 48]}
  Administrators can assign roles and enable application version checking to control access to the selected application.

- \textbf{Uploading and Deploying Hybrid Apps [page 49]}
  If the selected hybrid app uses the \textit{AppUpdate} plugin, you can activate a new version of the app.

- \textbf{Defining Client Policies [page 51]}
  Set policies related to client password and log management for mobile applications.

- \textbf{Uploading Client Resources [page 57]}
  Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Cloud Platform mobile service for development and operations. The administrator can modify the client resource bundle settings in Mobile Service for Development and Operations cockpit.

- \textbf{Defining Connectivity [page 58]}
  Define destinations for the selected application.

- \textbf{Defining Back-end Connections for Agentry [page 64]}
  Define back-end connections for the selected Agentry application after publishing the application. Before then, the Back End tab is blank. The application developer creates initial back-end connection settings during development, and then publishes the application to SAP Cloud Platform mobile service for development and operations. Once you publish the application, you must modify the initial back-end connection values for the SAP Cloud Platform mobile service for development and operations environment.

- \textbf{Configuring the Document Repository [page 68]}
  Enable access to the document repository for the selected application without registering. You’ll also be allowed to move documents and refresh the root folder.

- \textbf{Enabling JSON Storage [page 69]}
  Enable and manage persistent JSON storage for the selected application.

- \textbf{Defining Offline Settings for Applications [page 70]}
  Define offline settings for the selected application. Offline support enables client applications to access back-end data without a connection. When offline, applications access data from an offline store on the client. SAP Cloud Platform mobile service for development and operations moves data between the back end and the client offline store.

- \textbf{Defining Push Notifications [page 79]}
  Configure push-related settings for the selected application.

- \textbf{Using the Sample Back End [page 86]}
  A sample OData service is available for developers to use during development and testing. The sample OData service also supports delta tokens, which let you evaluate how delta tokens are handled in your test application.
Using the Cloud Build Service [page 90]
Build custom Fiori Clients for your Fiori Launchpad or individual SAP Fiori business apps, and enable them to use the SAP Mobile Platform SDK. Use the Cloud Build service to build the binary.

Defining Application Security [page 92]
Define the settings that control user authentication behavior for the selected application.

1.4.3.1.3.1 Defining Access Control

Administrators can assign roles and enable application version checking to control access to the selected application.

Context

At runtime, SAP Cloud Platform mobile service for development and operations verifies that the application users have the required roles. In addition, administrators can enable application version checking. Only the application client with the active version can run and access the application back-end.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid
2. Select an application, and then select Access Control under Assigned Features (or add it first).
3. To enable role checking for the application, select Enable Role Settings.
   For Roles, enter the name of a role that a user must be assigned to run the application. If multiple roles can run the application, enter the role names separated by commas. Users registering an application must be granted at least one of the roles configured in the access control policy or the registration fails with a response code of 403.
   If you want to enter multiple role names, you can:
   ○ Enter all the role names separated by commas, for example: "Developer, Sales, Manager".
   ○ Type role names separated by the ENTER key, for example: type "Developer" + ENTER key, type "Sales" + ENTER key, type "Manager" + ENTER key.
   All the role names you entered are listed in the multi-input field. You can review the list by using the left or right cursor keys, or by swiping to the left or right.
4. To enable application version checking, select Enable Application Versioning.
   You can add, update, or remove versions, and can choose the active version from the versions table list. Only the application client that is the active version can run and access the application back-end.
1.4.3.1.3.2 Uploading and Deploying Hybrid Apps

If the selected hybrid app uses the AppUpdate plugin, you can activate a new version of the app.

Prerequisites

A hybrid app package that:

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

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

- Is compressed into a standard .zip file for upload.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select App Update under Assigned Features (or add it first).
3. To import a new application or update an existing application version, click Upload.
   a. In the dialog, click Browse, and navigate to the directory that contains the hybrid app package.
   b. Select the package, and click Upload.

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

Table 17: Hybrid App Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Operating System</td>
<td>The operating system on which the application runs.</td>
</tr>
<tr>
<td>Revisions</td>
<td>Identifies the production version revision. A revision number is assigned to a newly uploaded hybrid app and incremented when a new version is uploaded.</td>
</tr>
</tbody>
</table>

i Note

When the hybrid app is deployed, the revision number is incremented.
<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Required Client Version</td>
<td>Identifies the hybrid SDK version that was used to develop the hybrid app, for example, 3.0.0.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>This version attribute is informational only; it is not used to determine whether device clients should receive a Web application update.</td>
</tr>
<tr>
<td>Development Version</td>
<td>Identifies the internal development version that was used to develop the hybrid app.</td>
</tr>
<tr>
<td>Uploaded By</td>
<td>The user who uploaded the hybrid app.</td>
</tr>
<tr>
<td>Last Modified</td>
<td>Date last modified.</td>
</tr>
<tr>
<td>State</td>
<td>State of the hybrid app version:</td>
</tr>
<tr>
<td></td>
<td>○ New – a newly uploaded version.</td>
</tr>
<tr>
<td></td>
<td>○ Staged – in testing. A user who is defined as a tester can download and test applications. See <em>Managing Registrations and Users</em>.</td>
</tr>
<tr>
<td></td>
<td>Once testing is complete, an administrator can promote a version to the Current state, so it becomes active. If testing fails, the administrator can change the state back to New.</td>
</tr>
<tr>
<td></td>
<td>○ Current – the version that is currently active.</td>
</tr>
</tbody>
</table>

**Table 18: Moving Applications Between States**

<table>
<thead>
<tr>
<th>Beginning State</th>
<th>Action</th>
<th>Ending State</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td>Click <em>Stage</em></td>
<td>Staged</td>
</tr>
<tr>
<td>New</td>
<td>Click <em>Deploy</em></td>
<td>Current</td>
</tr>
<tr>
<td>Staged</td>
<td>Click <em>Remove</em></td>
<td>New</td>
</tr>
<tr>
<td>Staged</td>
<td>Click <em>Deploy</em></td>
<td>Current</td>
</tr>
</tbody>
</table>

**Note**

Each platform can have an unlimited number of versions in the New state, but only one version in the Staged state, and one version in the Current state.

4. To deploy applications, select the check box for each application you want to deploy, and click *Deploy*.

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

For device-application users:
- If a hybrid app with the default version (revision = 0) connects to the server, the server downloads the full hybrid app.
- If a hybrid app with a version (revision = 1 or higher) connects to the server, the server calculates the difference between the user’s version and the new version, and downloads a patch containing only the required changes.
- If the application implements the *AppUpdate* plugin, the server checks for updates when the application starts or resumes. If the developer has made changes, *AppUpdate* detects them using the *www* folder content (the HTML-based content), and not with native plugins or changes made outside of
that folder. For changes made outside the www folder, the developer must post a new copy of the app to the application download site, or use Afaria to push the new app to all users.

5. To remove application versions that have been imported, but not yet deployed, select the check box for each application you want to remove, and click Remove.

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

Related Information

Managing Registrations and Users [page 128]

1.4.3.1.3.3 Defining Client Policies

Set policies related to client password and log management for mobile applications.

Defining Client Password Policy [page 52]
Define the client password policy used to unlock the DataVault, for the selected application. Application developers must add code to enforce the policy to the DataVault used by the application. An administrator enters the application password policy used to unlock the DataVault during application initialization.

Defining Client Log Policy [page 53]
Define the policy for uploading client logs to the database.

Defining Client Database Upload Policy [page 54]
Define the policy for uploading database files from the device client.

Defining Usage Report Policy [page 55]
Enable the server to upload application-specific usage statistics and reports from mobile devices.

Defining Feature Restriction Policy [page 55]
Feature restriction policies enable you to control feature access for the selected application. Set these policies in Mobile Service for Development and Operations cockpit. You can add, allow, restrict, edit, or delete features.
1.4.3.1.3.3.1 Defining Client Password Policy

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

Context

The client password policy applies only to the application password that unlocks the DataVault during application initialization; it affects neither SAP Cloud Platform mobile service for development and operations security profiles nor the back-end security systems with which it integrates. Password policies for back-end security systems are administered by customer information technology departments using native security administration tools.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications >> Native/Hybrid
2. Select an application, and then select Client Policies under Assigned Features (or add it first)
3. Under Passcode Policy, select Enable Passcode Policy, and enter:

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Expiration Time Frame Days</td>
<td>0</td>
<td>The number of days a password remains valid. The default value, 0, means the password never expires.</td>
</tr>
<tr>
<td>Minimum Length</td>
<td>8</td>
<td>The minimum password length.</td>
</tr>
<tr>
<td>Retry Limit</td>
<td>10</td>
<td>The number of retries allowed when entering an incorrect password. After this number of retries, the client is locked out, the DataVault and all its contents are permanently deleted, the application is unusable, and encrypted application data is inaccessible.</td>
</tr>
<tr>
<td>Minimum Number of Unique Characters</td>
<td>0</td>
<td>The minimum number of unique characters required in the password.</td>
</tr>
<tr>
<td>Lock Timeout</td>
<td>300</td>
<td>The number of seconds the DataVault remains unlocked within an application, before the user must reenter his or her default password to continue using the application (similar to a screen-saver feature).</td>
</tr>
<tr>
<td>Default Passcode Allowed</td>
<td>Disabled</td>
<td>If enabled, a default password can be generated by the DataVault; from the user’s point of view, this turns off the password.</td>
</tr>
</tbody>
</table>
### 1.4.3.1.3.3.2 Defining Client Log Policy

Define the policy for uploading client logs to the database.

**Context**

Log settings for an application are applied to all installed instances of an application. You can override those application-level settings for a particular device, by changing the settings for a particular “deviceID”.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select [Mobile Applications](#) > [Native/ Hybrid](#).
2. Select an application, and then select [Client Policies](#) under [Assigned Features](#) (or add it first).
3. Under **Log Policy**, select **Enable Log Policy**.
4. Select the log level (default is None):

<table>
<thead>
<tr>
<th>Log Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>For tracing execution flow. Used, for example, in the context of entering and leaving a method, looping, and branching operations. (Not applicable to the offline logging component.)</td>
</tr>
<tr>
<td>Debug</td>
<td>For debugging purposes, includes extensive and low-level information.</td>
</tr>
<tr>
<td>Info</td>
<td>Informational text, used mostly for echoing what has been performed.</td>
</tr>
<tr>
<td>Warn</td>
<td>The application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator.</td>
</tr>
<tr>
<td>Error</td>
<td>The application can recover from the error, but cannot fulfill the task due to the error.</td>
</tr>
<tr>
<td>Log Level</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Fatal</td>
<td>The application cannot recover from the error, and the severe situation causes fatal termination.</td>
</tr>
</tbody>
</table>

5. Select the time period after which logs are deleted from the database.

You should retrieve these settings at runtime, for more information see Uploading and Viewing Client Logs. By default, log files exist in the database for seven days. To read the logs, see Viewing Event Logs [page 165].

6. Click Save.

1.4.3.1.3.3.3 Defining Client Database Upload Policy

Define the policy for uploading database files from the device client.

**Context**

Enable the client database file upload policy for an individual application to troubleshoot problems on the client side. The application must be configured by the developer to allow this feature; the administrator must enable the policy; and the application users must upload the database files from the client device. Once the files are uploaded, the developer can troubleshoot problems. The database files are deleted once the length of time specified by the administrator has elapsed.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid
2. Select an application, and then select Client Policies under Assigned Features (or add it first)
3. Under Database Upload Policy, enable Database Upload Policy, and enter:

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Delete Database After</td>
<td>10080 seconds</td>
<td>The time to elapse before the database files are deleted automatically. This security measure protects the device user, but be sure to allow enough time for the developer to perform the troubleshooting or analysis.</td>
</tr>
<tr>
<td>Property</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Maximum Database Size</td>
<td>32 MB</td>
<td>The maximum size allowed for the database files.</td>
</tr>
</tbody>
</table>

4. Click Save.
5. Once the policy is enabled, the device user can upload the local database files for the developer to analyze or troubleshoot.

### 1.4.3.1.3.3.4 Defining Usage Report Policy

Enable the server to upload application-specific usage statistics and reports from mobile devices.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid
2. Select an application, and then select Client Policies under Assigned Features (or add it first)
3. Under Usage Report Policy:
   a. To enable the server to accept records for an application, select Enable Usage Report Policy.
   b. In Expiration Time Frame, enter the number of days, after which usage reports expire.

**Next Steps**

To view usage information, select Analytics.

### 1.4.3.1.3.3.5 Defining Feature Restriction Policy

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

**Context**

When you edit a hybrid app in Mobile Service for Development and Operations cockpit, available feature plugins are listed on the Client Policies screen. Feature plugins are typically JavaScript APIs that provide...
access to the native APIs of the mobile device (implemented as Apache Cordova plugins, for example, Camera, Calendar, and Push). You can indicate features that should be restricted from the user.

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

- Invalidates the native side of the plugin
- Changes the namespace of the plugin to null

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

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

**Note**

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

### Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications ➤ Native/Hybrid**

2. Select an application, and then select **Client Policies** under **Assigned Features** (or add it first)

3. Under Feature Restriction Policies, you see the current status of feature restriction policies.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plugin</td>
<td>A list of feature plugins that are available with the application, such as Camera, Calendar, and Push.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique identifier for the plugin.</td>
</tr>
<tr>
<td>Allowed</td>
<td>Indicates whether the feature is allowed or restricted.</td>
</tr>
<tr>
<td></td>
<td>○ Select if the feature is allowed (Yes).</td>
</tr>
<tr>
<td></td>
<td>○ Unselect if the feature is restricted (No).</td>
</tr>
</tbody>
</table>

4. (Optional) Add feature restrictions for the selected application, and then save.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique feature name.</td>
</tr>
<tr>
<td>Plugin</td>
<td>A feature plugin that is available with the application, such as Camera, Calendar, and Push.</td>
</tr>
<tr>
<td>Plugin Name</td>
<td>Plugin name.</td>
</tr>
</tbody>
</table>
### Field Description

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>A feature plugin description, such as Cordova Camera Plugin, Cordova Contacts Plugin, and SAP Push Plugin.</td>
</tr>
<tr>
<td>JavaScript Module</td>
<td>A comma-separated list of all JavaScript modules that are used by this plugin. The JavaScript Module value is the JavaScript API that is used to invoke the plugin.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique identifier for the plugin. The value comes from the cordova_plugins.js file, which appears in the project after you add a plugin (&quot;pluginId&quot;).</td>
</tr>
<tr>
<td>Allowed</td>
<td>Indicates whether the feature is allowed or restricted. By default, features are allowed.</td>
</tr>
</tbody>
</table>

5. (Optional) To remove a plugin from the application, select it, click delete, and confirm.

### 1.4.3.1.3.4 Uploading Client Resources

Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Cloud Platform mobile service for development and operations. The administrator can modify the client resource bundle settings in Mobile Service for Development and Operations cockpit.

### Context

Keep in mind these resource bundle guidelines:

- **Supportability** – the resource bundle can be of any type (.pdf, .xls, .xml, or any other extension), with no restrictions.
- **Size** – the resource bundle is restricted in size. The maximum size is 64 MB. For best performance, SAP recommends a maximum size of 1MB. For larger files, work with an application developer for performance issues.
- **Default resource bundle** – the first resource bundle that you upload is considered to be the default. After that, you can upload additional versions of the bundle, but only one can be the default. You can delete obsolete resource bundle versions.
- **URL for the default resource bundle** – https://<hmc base url>/bundles/<ApplicationName>/.
- **URL to access other resource bundles** – https://<hmc base url>/bundles/<ApplicationName>/<BundleName>:<BundleVersion>.

### Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid
2. Select an application, and then select Client Resources under Assigned Features (or add it first)
3. Click the Upload Client Resource icon to create a new Client resource:
Bundle Name – provide a name to identify the resource.
Version – provide a version number.
Upload Client Resource – click Browse, select the file, and confirm.

4. To define a client resource bundle as the default, select it, and click Save.

1.4.3.1.3.5 Defining Connectivity

Define destinations for the selected application.

Context

A destination is a connection to a data source.

SAP Cloud Platform mobile service for development and operations supports one primary endpoint per application ID. However, an administrator can create multiple secondary endpoints for services that an application uses; these secondary endpoints are treated as proxy connections. For applications that access Web services containing relative URLs, add the relative paths to enable the product to handle requests correctly.

In Mobile Service for Development and Operations cockpit, you can view the properties of Fiori applications and connections that were developed using SAP Cloud Platform mobile service for app and device management and imported into SAP Cloud Platform mobile service for development and operations, but you cannot edit their properties; input fields and buttons are disabled or hidden. A default Fiori Client destination, named com.sap.fiorimobile.con.Fiori_Cloud_Edition, is created automatically for the Fiori Destination.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid.
2. Select an application, and then select Connectivity under Assigned Features (or add it first).

View current mobile, Fiori, and cloud platform destinations for the selected application.

Table 23: Destinations

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The destination name.</td>
</tr>
</tbody>
</table>
### Field | Value
--- | ---
Platform Destination Name | For Cloud Platform Destination types, this is the original cloud platform destination name (it may differ from the destination name used in Mobile Service for Development and Operations cockpit).
For Mobile Destination types, this is not used (Not Applicable).
For Fiori Destination types, this is not used (Not Applicable).
URL | The destination URL.
Proxy Type | The proxy type used for the destination – Internet, or On Premise (Cloud Connector).
SSO Mechanism/Authentication | The single sign-on or authentication security methods employed for the destination.
Actions | The actions available, such as test an OData application destination, ping a back-end connection, or delete a connection.

3. (Optional) Select the new New icon and use the Create Destination dialog to create a new destination. See [Creating a Destination](#) for details.

4. (Optional) Select the Add icon to add an existing destination. Select a destination from the list, and click OK.

**Note**

If you selected Fiori Destination, a default Fiori Client destination, named com.sap.fiorimobile.con.Fiori_Cloud_Edition, is available by default.

5. Select a row to view its settings in the Destination Overview.

- **Rewrite Modes** [page 60]
  
  For application back-end connections, the rewrite mode defines how the mobile service for development and operations handles request and response messages.

- **Configuring Application-to-Application SSO Authentication** [page 63]
  
  The application-to-application single sign-on (SSO) authentication type allows SAP Cloud Platform mobile service for development and operations to propagate the identity of the logged-in user to another application, which is consumed (deployed or subscribed) in the same SAP Cloud Platform account.

- **Testing an OData Application Destination** [page 64]
  
  You can now test destination links for OData applications from the Connections window, via a test wizard.
1.4.3.1.3.5.1 Rewrite Modes

For application back-end connections, the rewrite mode defines how the mobile service for development and operations handles request and response messages.

To enable applications that use external back ends to run offline, select one of the supported rewrite modes.

Rewrite URL on Back End

The back end rewrites the URLs. The mobile service for development and operations forwards the host name and port to the back end in an HTTP header, and the back end creates the URL to retrieve back-end resources.

To expose the full URL to clients, the mobile service for development and operations passes the endpoint in the X-SMP-ENDPOINTNAME header, for example:

- **URL exposed to clients** – http://<smphost>:<port>/sap/opu/odata/sap/FINCUSTFACTSHEET/
- **URL format for Web applications** – https://<host>/<back-end path>?X-SMP-APPID=<applicationID>, for example:
  
  ```
  ```

To transparently register Web-type applications, specify the application ID as the value of the X-SMP-APPID parameter in the URL. Keep in mind:

- You can specify any application ID as the value of X-SMP-APPID.
- If the application does not exist or has no valid endpoint in the specified path, it is not called.

Considerations

- The base path of the URL must correspond to the path of the back-end URL. For other rewrite modes, the base path must contain the application ID (as in the example above).
- If you change the value to or from Rewrite URL on Back End, inform the application developer of the change. He or she must update the application base URL accordingly, for both online and offline mobile application scenarios.
- If you change the rewrite value, you must also reconfigure the mobile application.
- Depending on other applications configured in your account, it may be impossible to detect from the URL which application is targeted; therefore, SAP recommends that you append the application ID as a URL parameter when you open the application in a browser, for example: https://server/webapp?X-SMP-APPID=webapp.
If the URL needs additional query parameters, add them before the application ID, for example: https://server/webapp?other=parameter&needed=true&X-SMP-APPID=webapp.

**Rewrite via Cloud Platform App**

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

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

**Rewrite URL**

In request and response messages, the mobile service for development and operations replaces all back-end URLs with the mobile service URL.

The rewrite URL format for Web-type applications is https://<host>/<applicationID>?X-SMP-APPID=<applicationID>.

**No Rewriting**

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

**Note**

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

The URL format for Web applications is https://<host>/<applicationID>?X-SMP-APPID=<applicationID>.

**Custom Rewrite URL**

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

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

- **Search For** – string to find. To facilitate searching, you can use placeholder variables, for example, to find the current application, enter `$(SMP_APPID)`.
- **Replace With** – replacement string. For example, you can convert an absolute URL to a relative URL by replacing "http://host" with an empty string.
• **Match Case** – whether the case of the search string must match exactly.
• **Regular Expression** – allow regular expressions in the strings.

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

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

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

To test a Custom Rewrite URL configuration for either inbound or outbound rules:
1. Select the configuration, and click Test.
2. In the Test Rewrite Rules dialog, enter the Input string.
3. Click Rewrite. The replacement string appears under Output.

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

**Related Information**

Defining Connectivity [page 58]
Creating a Destination [page 135]
1.4.3.1.3.5.2 Configuring Application-to-Application SSO Authentication

The application-to-application single sign-on (SSO) authentication type allows SAP Cloud Platform mobile service for development and operations to propagate the identity of the logged-in user to another application, which is consumed (deployed or subscribed) in the same SAP Cloud Platform account.

Prerequisites

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

Note

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

Context

The user identity is propagated to the application specified in the URL, which you can configure in Mobile Service for Development and Operations cockpit.

Procedure

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

You can now test destination links for OData applications from the Connections window, via a test wizard.

Prerequisites

The destination’s SSO Basic Auth must be configured with a technical user. If the icon does not appear, the OData application destination is not configured correctly.

Context

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

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid
2. Select an application, and then select Connectivity under Assigned Features (or add it first)
3. Under Actions, click the OData application test destination icon .
   The OData Application Destination Test wizard appears.
4. On the first page, view the basic information about the application and destination. You can enter the relative OData path of the destination URL, and check the OData metadata definition.
5. On the second page, view the OData service entities, the properties and data of the entities; and the JSON raw data and XML raw data of each entity.

1.4.3.1.3.6 Defining Back-end Connections for Agentry

Define back-end connections for the selected Agentry application after publishing the application. Before then, the Back End tab is blank. The application developer creates initial back-end connection settings during development, and then publishes the application to SAP Cloud Platform mobile service for development and operations. Once you publish the application, you must modify the initial back-end connection values for the SAP Cloud Platform mobile service for development and operations environment.

Communication between SAP Cloud Platform mobile service for development and operations and the back-end system is represented within the application by the back-end connection type. You can use the Java Virtual Machine Connection to connect to SAP from Agentry via custom Java code.
Before your Agent application can communicate with some back-end systems, you need to set up Agent application host server connectivity.

### Prerequisites

- Obtain and record the back-end identifier for the Java Virtual Machine system connection. This is the name or number defined in the Agent Editor.
- Obtain and record the host system network name of the Java application server to which SAP Cloud Platform mobile service for development and operations is to connect.
- Obtain and record the application server name of the Java interface, where applicable.
- Identify any .jar or .class files that contain business objects that the application being implemented may use, and make them accessible to SAP Cloud Platform mobile service for development and operations. These files are normally provided with the back-end system with which the SAP Cloud Platform mobile service for development and operations is synchronizing.
- Determine whether users are authenticated using the JVM system connection. If they are, they cannot synchronize unless they have first been successfully authenticated. Proper user credentials must be established within the back-end system before users are allowed access. This may be done before or after configuring the back-end connection in Mobile Service for Development and Operations cockpit.
Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications Configure Back End.

2. Under JAVA, edit JAVA-related settings. These configuration options are for a single Java Virtual Machine system connection, and control the connection behavior between the server and a Java interface. There may be multiple Java-<n> sections within the agentyr.ini file. One must exist for each Java Virtual Machine system connection definition within the application. The <n> portion of the section name must be replaced with the ID value generated by the Editor for the system connection definition.

<table>
<thead>
<tr>
<th>Definitions</th>
<th>Values</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class Path</td>
<td>Valid fully qualified paths.</td>
<td>Contains multiple path values that specify the location of different Java resources. This can include .jar files used by the application. Each path must be fully qualified. Each entry must be separated by a semicolon. All paths for this setting are added to the system variable CLASSPATH of the server’s host system. Requires application restart.</td>
</tr>
<tr>
<td>Constants File</td>
<td>Valid file name or path and file name. Default: none.</td>
<td>Name and location of the .ini file, relative to the server, that contains constants that can be accessed from the Java code via the SessionData.</td>
</tr>
<tr>
<td>Delete Source (Deprecated)</td>
<td>Boolean value of true or false. Default: false</td>
<td>Controls whether the files stored in Source Directory are deleted after being successfully compiled by the JVM. This may be necessary in a development environment if the Java source files are modified outside of the Agentry Editor. In development mode, the Agentry Editor can push Java source files to the server for the server to compile. Requires application restart.</td>
</tr>
<tr>
<td>Enable Authentication</td>
<td>Boolean value of true or false. Default: true</td>
<td>Whether users are authenticated against the back-end system for this system connection. At least one system connection within the application must perform user authentication.</td>
</tr>
<tr>
<td>Enable Previous User Authentication</td>
<td>Boolean value of true or false. Default: true</td>
<td>Whether previous users are authenticated against the back-end system for this system connection. This authentication occurs when a user change occurs on the client.</td>
</tr>
<tr>
<td>Name</td>
<td>Text value.</td>
<td>Any text value that identifies the system connection in log files and other areas. Set each connection name to a unique value. Requires application restart.</td>
</tr>
<tr>
<td>Output Directory (Deprecated)</td>
<td>Valid path value, relative to the server.</td>
<td>The location where the compiled .class files produced by the JVM are to be stored for execution. The standard is to use the Java subfolder in the server’s installation folder;</td>
</tr>
<tr>
<td>Definitions</td>
<td>Values</td>
<td>Descriptions</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>--------------</td>
</tr>
<tr>
<td>however, you can use another location. This path value you enter here must also be a part of the system’s CLASSPATH (either using a system variable, or by adding the value to the classPath configuration option). This only applies to applications running in development mode: in development mode the Agentry Editor can push Java source files to the server for the server to compile. Requires application restart.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Print Business Logic Stack Trace</td>
<td>Boolean value of true or false. Default: false</td>
<td>Setting this option to true results in messages generated by the AJ-API exception class <code>JavaBusinessLogicError</code> being printed to the <code>events.log</code> file produced by the server. Requires application restart.</td>
</tr>
<tr>
<td>Print Stack Trace</td>
<td>Boolean value of true or false. Default: false</td>
<td>Setting this option to true results in messages generated by the exception stack trace being printed to the <code>events.log</code> file produced by the server. Requires application restart.</td>
</tr>
<tr>
<td>Scripts Path</td>
<td>Valid file name or path and file name. Default: none.</td>
<td>Name and location of the scripts file, relative to the server. This is the directory in which Agentry Editor places Java source code, when the source code for a Java class is entered directly into the Editor to define a steplet or similar. Requires application restart.</td>
</tr>
<tr>
<td>Server Class</td>
<td>Valid Java Server Class extension.</td>
<td>The application-specific extension of the AJ-API <code>syclo.agentry.server</code> class. This option is automatically set when this API class has been extended by a server class for the application. If no such class exists, the default <code>syclo.agentry.server</code> class is assumed and should not be set for this configuration option. Requires application restart.</td>
</tr>
<tr>
<td>Source Directory (Deprecated)</td>
<td>Valid path value, relative to the server.</td>
<td>This directory can be used for all other source code on which the Scripts Path might depend. This is the location of the <code>.java</code> files for the <code>Steplet</code>, <code>ComplexTable</code>, and <code>DataTable</code> classes. Files are placed here before they are compiled by the JVM. This is not where the Java project files reside that are related to the application. The <code>.java</code> files are written to this location by the server at runtime. Their contents are based on the definitions in which they are contained. This only applies to applications running in development mode: in development mode the Agentry Editor can push Java source files to the server for the server to compile.</td>
</tr>
</tbody>
</table>
### Configuring the Document Repository

Enable access to the document repository for the selected application without registering. You'll also be allowed to move documents and refresh the root folder.

**Prerequisites**

The document repository service can only be used by applications with security configured for NONE or BASIC (and not SAML, the default). See *Defining Application Security [page 92]* to learn more about security configurations.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications > Native/Hybrid**.
2. Click **New**.
3. Select **Config Template** for the new application.
4. Enter an **ID** and a **Name** for the application, and click **Save**.
   
   **Note**

   To configure the document repository for an existing application, select the required application.

5. In the **Info** tab, click the **+** icon in the **Assigned Features** section.
6. In the Add feature window, select **Document Repository** and click **OK**.
7. In the **Configuration** tab, select **edit** icon.
8. Enter the **New Folder Name** and click **OK**.
9. To import a file into the folder, select the **arrow up** icon.
10. Select the file to import, and click **Import**.

   **Note**
   To overwrite an existing file, select **Overwrite existing file?**.

11. To open an existing file, select the 📑 icon, and click **OK**.
12. To move a file or folder from one folder to another, select the file or folder and click **Move** icon.
   1. Select a target folder and click **OK**.
13. To download an existing file, select the 🔗 icon.
14. To delete an existing file or folder, select the required folder or file and click the ⚑ icon.

   **Note**
   You can delete a folder only if they are empty.

15. To rename an existing file or folder, click **Rename**.

### 1.4.3.1.3.8 Enabling JSON Storage

Enable and manage persistent JSON storage for the selected application.

**Context**

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

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** ➔ **Native/Hybrid**
2. Select an application, and then select **JSON Storage** under **Assigned Features** (or add it first)

   Under **Storage Level** you can view three levels of storage in a tree format – application, user, and device.
When you select a tree node, the storage JSON appears in the **JSON Editor** to the right. Keep in mind:

- The tree only lists users that use JSON storage, or that have one or more registered devices that use JSON storage.
- The tree lists all user devices, whether the device has storage JSON.

3. (Optional) Under **Storage Level**, use the filters to adjust the storage entries you see for the application. You can limit or expand the number of entries to retrieve, search for a user by name, or search the tree for a specific value. You can expand or collapse the tree of storage entries. You can add a user (for example, if a user does not appear in the list, since he or she does not currently have storage), or delete a user.

4. Select a level in the tree to manage storage for the application.

<table>
<thead>
<tr>
<th>Storage Service</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Level</td>
<td>View and modify application-level storage.</td>
</tr>
<tr>
<td>User Level</td>
<td>View and modify user-level storage.</td>
</tr>
<tr>
<td>Device Level</td>
<td>View and modify device-level storage.</td>
</tr>
</tbody>
</table>

5. (Optional) Select an Application Level object to manage the storage service at the application-level. The application ID appears, as well as a **Role Definition** section.
   - For **Role Definition**, assign one or more roles to the application for reading and writing. Enter one role name at a time followed by **Enter** to create a list of roles.
   - In **JSON Editor** you can preview code in JSON format, or modify the JSON format. Use **Beautify** to format code, and **Full Preview** to view the full application level JSON storage.

6. (Optional) Select a User Level object to manage storage service at the user-level. The user ID appears.
   In **JSON Editor** you can preview code in JSON format, or modify the JSON format. Use **Beautify** to format code, and **Full Preview** to view the full user level JSON. The full JSON is a combination of user level and application level JSON.

7. (Optional) Select a Device Level object to manage the storage service at the device-level. The device ID appears.
   In **JSON Editor** you can preview code in JSON format, or modify the JSON format. Use **Beautify** to format code, and **Full Preview** to view the full device level JSON. The JSON is a combination of device level, user level, and application level JSON.

8. (Optional) Select an object, and click ![Trash Can Icon] to remove it.

9. Click **Save** at any time to save your changes, or **Reset** to discard all unsaved changes.

### 1.4.3.1.3.9 Defining Offline Settings for Applications

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

Context

The destination settings determine how SAP Cloud Platform mobile service for development and operations creates the initial offline store database on the client, and how it processes requests for updates from the back end. Define offline back-end connection settings for an application by importing a configuration (.ini) file that has been prepared by a developer. You cannot update the settings using Mobile Service for Development and Operations cockpit. To adjust any settings, remove the current configuration, update the configuration file, and reimport the file. Before updating this file, confer with a developer. See these sections in the Native OData App Development and SAP Cloud Platform SDK for iOS documentation:

- iOS Applications
- Android Applications
- Windows Applications
- Developing Offline Applications

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid.
2. Select an application.
3. In the Info tab, click the icon in the Assigned Features section.
4. In the Add feature window, select Connectivity and click OK.
5. To add an existing destination, in the Configuration tab, select the icon.
6. Select the destinations you want to add, and click OK.
7. To test an OData application destination, select the icon.
8. To check whether a destination is active, elect Ping.
9. To delete a destination, select the icon.
10. Return to the application.
11. In the Info tab of the application, click the icon in the Assigned Features section.
12. In the Add feature window, select Offline and click OK.
13. Click Offline Settings. You see the list of configured destinations.
14. To import an offline configuration, select the icon, browse to and select a configuration file. Only .ini files can be imported. When you import settings, the state of the offline configuration changes to Configured.
15. To view the uploaded client databases, select the **Client Databases** tab.

You see the list of existing client databases that are available in offline settings. You can also view the Unique ID, Device ID, Created By, File Size, Creation Data, and the Actions you can perform for each database.

16. (Optional) To download an existing database, click the ✗ icon.

17. (Optional) To delete an existing database, select the corresponding check box, and click the-trash icon.

**Application Configuration File**

Use the application configuration file to configure OData endpoints and defining queries that determine the data that populates the application database when it is created. Use the cockpit to import the application configuration file.

### 1.4.3.1.3.9.1 Application Configuration File

Use the application configuration file to configure OData endpoints and defining queries that determine the data that populates the application database when it is created. Use the cockpit to import the application configuration file.

An application configuration file consists of one or more OData endpoints, each of which can be optionally followed by one or more defining queries.

Each application configuration file represents a single application. All settings have default values, so an application configuration file is necessary only if you want behavior other than what is provided by the defaults.

Endpoints are defined as follows:

```ini
[endpoint]
name=YourConnectionName
prepopulate_offline_db=Y|N|SHARED-ONLY
prepopulate_offline_db_interval=NNN
indexed_type=<namespace>.<entity type name1>: Property1A [ASC|DESC][, Property1B [ASC|DESC], ...]
indexed_type=<namespace>.<entity type name2>: Property2A [ASC|DESC][, Property2B [ASC|DESC], ...]
request_format=application/json;q=1,application/atom+xml;q=0.5
delta_request_format=application/atom+xml
offline_db_collation=UTF8BIN
case_sensitive_offline_db=Y|N
json_datetimeoffset_in_utc=Y|N
local_change_expiry=NNN
allow_omitting_max_length_facet=Y|N
content_id_header_location=mime|operation
max_delta_resends=NNN
batch_all_defining_requests=Y|N
refresh_in_order=Y|N
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Name</td>
<td>(Required) The name of the endpoint, defined as a <strong>Connection Name</strong> in the cockpit.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Valid Values</td>
<td>Notes</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>prepopulate_offline_db</td>
<td>Y/N</td>
<td>The type of data that is included in the initial database.</td>
</tr>
<tr>
<td></td>
<td>SHARED-ONLY</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Shared-only</strong></td>
</tr>
<tr>
<td>prepopulate_offline_db_interval</td>
<td>NNN</td>
<td>If a download refresh requires a new database, the shared data is refreshed based on the interval, in minutes, specified here. The default is 1440 minutes (one day). This option applies only when prepopulate_offline_db is set to <strong>Y</strong> or <strong>SHARED-ONLY</strong> and there is at least one defining query defining request with <strong>is_shared_data=Y</strong>.</td>
</tr>
<tr>
<td>indexed_type</td>
<td>&lt;namespace&gt;..&lt;en-</td>
<td>Each indexed_type defined causes the client database to have an index for that type. Defining indexes can improve client query performance. However, defining too many indexes can slow the performance of downloads refreshes and updates. Specify multiple indexed_type entries by listing one entry per line. Specify either the primitive property name, or a complex type path expression that ends with a primitive property.</td>
</tr>
<tr>
<td></td>
<td>tity type name&gt;:</td>
<td></td>
</tr>
<tr>
<td></td>
<td>PropertyName [ASC</td>
<td>DESC]</td>
</tr>
<tr>
<td>request_format</td>
<td>atom</td>
<td>json</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

**SAP Cloud Platform Mobile Service for Development and Operations**

**SAP Cloud Platform Mobile Service for Development and Operations**

**PUBLIC** 73
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Notes</th>
</tr>
</thead>
</table>
| delta_request_format        | atom|json| any valid value            | The format to use when communicating with the OData producer. This property applies only when the offline OData service is making delta requests to the OData producer.  
  atom | Short for application/atom+xml.  
  json | Short for application/json.  
  Any valid value | Specify any valid value for an Accept header.  
  The default is: application/atom+xml |
| offline_db_collation        | Any collation supported by UltraLite | The collation for the database.  
  The default is: UTF8BIN |
| case_sensitive_offline_db   | Y|N                | Whether the database is case-sensitive.  
  The default is: case_sensitive_offline_db=Y |
| json_datetimeoffset_in_utc  | Y|N                | Whether the OData back end specifies or accepts the ticks portion of a JSON-formatted Edm.datetimeoffset in UTC or in the time zone specified by the offset portion.  
  Y | The OData back end accepts the ticks portion of a JSON-formatted Edm.datetimeoffset in UTC.  
  N | The OData back end accepts the ticks portion of a JSON-formatted Edm.datetimeoffset in the time zone that is specified in the offset portion.  
  The default is: json_datetimeoffset_in_utc=Y |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>local_change_expiry</td>
<td>NNN</td>
<td>Local data might persist in the local store after it has been flushed to the back end. To prevent this, specify an expiry interval, in hours, for local data. When this option is omitted from the configuration file, local changes never expire.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Negative value</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Zero</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Positive value</strong></td>
</tr>
<tr>
<td>allow_omitting_max_length_facet</td>
<td>Y</td>
<td>N</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Y</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>N</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>The default is:</td>
</tr>
</tbody>
</table>

**Note**

This feature is intended to be used for development purposes only as it does not provide fine-tuned control over the values chosen. In a production environment, ensure MaxLength facets for Edm.String and Edm.Binary key properties are included in the OData metadata.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Valid Values</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>content_id_header_location</td>
<td>mime</td>
<td>operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mime</td>
</tr>
<tr>
<td></td>
<td></td>
<td>operation</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>content_id_header_location=mime</td>
</tr>
<tr>
<td>max_delta_resends</td>
<td>NNN</td>
<td>When multiple defining queries defining requests are delta-enabled, the delta requests are automatically and repeatedly sent to the back end until all delta responses are empty, to ensure data consistency. Use this option to configure the number of times a delta link is resent during a download refresh. The default is 5.</td>
</tr>
<tr>
<td>batch_all_defining_requests</td>
<td>Y/N</td>
<td>Defined under [endpoint] level. If Y, all defining requests under the endpoint are sent in one batch operation. The default is N.</td>
</tr>
<tr>
<td>refresh_in_order</td>
<td>Y/N</td>
<td>The order in which to refresh defining requests. When this value is Y, the defining requests are queried in the order in which they are defined in the configure file. When the value is N (the default), they are queried in the order of refresh requests.</td>
</tr>
</tbody>
</table>

Create defining queries defining requests as follows:

```json
[defining_request
 Name=DefiningRequestName
 is_shared_data=N|Y
 download_refresh_interval=NNN
 track_deltas=AUTO|ALWAYS|NEVER
 delta_token_lifetime=NNN
]
```

Currently, only defining queries defining requests where the resource path identifies an entity set or entity type instance are supported.
Table 25:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>DefiningRequestName</td>
<td><em>(Required)</em> The name of the defining query defining request. A defining query defining request is an OData read request that targets the OData endpoint that is associated with the offline store, and retrieves a subset of the OData endpoint data. Defining queries defining requests are defined in the ODataOfflineStoreOptions object of the OData Offline API. The application populates the “definingRequests” parameter.</td>
</tr>
<tr>
<td>is_shared_data</td>
<td>N</td>
<td>Y</td>
</tr>
<tr>
<td>downloadrefresh_interval</td>
<td>NNN</td>
<td>The interval time, in minutes, between downloads refreshes of the shared data. The default is 15 minutes.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Value</td>
<td>Notes</td>
</tr>
<tr>
<td>-------------------</td>
<td>-------</td>
<td>-------</td>
</tr>
<tr>
<td>track_deltas</td>
<td>AUTO</td>
<td>ALWAYS</td>
</tr>
<tr>
<td></td>
<td>Auto</td>
<td>If the OData producer supports delta change tracking on the defining query defining request, the delta links returned are used when downloading the data. If the OData producer does not support delta change tracking on the defining query defining request, the server tracks the changes so that only changed data is exchanged between the client and the server. This is the default value.</td>
</tr>
<tr>
<td></td>
<td>Never</td>
<td>Sets the server to never track delta changes.</td>
</tr>
<tr>
<td></td>
<td>Always</td>
<td>Sets the server to always track delta changes. This option is equivalent to Auto if the OData producer does not support delta change tracking on the defining query defining request.</td>
</tr>
<tr>
<td>delta_token_lifetime</td>
<td>NNN</td>
<td>The time, in minutes, until the OData delta token expires. The default is 72000 minutes (50 days). While a token is valid, only changed data is downloaded and refreshed on the client database. When the delta token expires, the entire client database is replaced.</td>
</tr>
</tbody>
</table>

Create defining_request_group as follows:

```plaintext
[defining_request_group]
defining_requests=req1,req2,req3
[defining_request_group]
defining_requests=req4,req5,req6
```
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>defining_requests</td>
<td>Comma separated Defining Requests name</td>
<td>Defining requests specified at the same defining_request_group are sent in one batch operation.</td>
</tr>
</tbody>
</table>

- When `defining_request_group` is defined, you cannot set `batch_all_defining_requests` to Y.
- When `batch_all_defining_requests` is set to N, if the existing defining request does not belong to any group, then it is still requested individually.

### 1.4.3.1.3.10 Defining Push Notifications

Configure push-related settings for the selected application.

Two channels are available for sending push notifications:

- **Native push** – the push listener service provided with SAP Cloud Platform mobile service for development and operations allows back-end systems to send native notifications to devices. Application developers must enable push notification code in applications. You can also use the native push mechanism to push notifications to a subset of users.
- **Text messages (SMS)** – in addition, you have the option to push SMS text messages to devices applications, instead, or in addition to, native push. Device users must provide a telephone number (the Mobile Station International Subscriber Directory Number, or MSISDN) when they register the application. Users may be charged for text messages.

### Google Cloud Messaging Canonical IDs

For Google Cloud Messaging (GCM) clients, canonical IDs prevent problems that could occur if a client application inadvertently triggers multiple registrations for the same device; for example, the device could receive duplicate messages.

If a GCM client application sends a message that contains an old registration ID, GCM processes the request and inserts the canonical ID into the `registration_id` field of the response.

SAP Cloud Platform mobile service for development and operations:

- Replace the old registration ID that is stored for the device with the canonical ID.
- Use the canonical ID for sending messages to the device.
- Log a customer event to inform the client of updated or deleted device registrations that result from managing canonical registration IDs.

### Related Information

- Push Text Messages (SMS) [page 81]
Enable SMS text messaging, and configure settings.

**Predefined Push Configuration** [page 81]
(SAP Content to Go and Fiori Client) Enable or disable preconfigured push settings.

**Custom Push Provider** [page 82]
(Optional) Enable the custom push provider to manage push notifications for the application at the device platform level, rather than for all device platforms.

**Android Push Notifications** [page 83]
To enable client applications to receive Google Cloud Messaging (GCM) notifications, configure Android push notifications for the selected application.

**Apple Push Notifications** [page 83]
To enable client applications to receive APNS notifications, configure Apple push notifications for the selected application.

**BlackBerry Push Notifications** [page 84]
To enable client applications to receive BES/BIS notifications, configure BlackBerry push notifications for the selected application.

**Windows Push Notifications** [page 85]
To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

**Windows Phone Push Notifications** [page 85]
To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows phone users running mobile applications, configure Microsoft push notification services (MPNS) for the selected application.

**Related Information**

- Defining Applications [page 43]
- Defining Connectivity [page 58]
- Enabling Applications to Discover Configurations [page 97]
- Managing Applications [page 121]
- Managing Registrations and Users [page 128]
- Viewing Server Data Report [page 159]
1.4.3.1.3.10.1 Push Text Messages (SMS)

Enable SMS text messaging, and configure settings.

Prerequisites

You'll need to provide several SMS provider values, including the login name, password, and originating address of the SAP SMS 365 provider account.

Context

You have the option to push SMS text messages to devices applications, as well as to push native notifications. This describes how to configure the SMS message channel for the application.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)
3. Under Push Text Messages (SMS), enable push text messages for the application, and configure settings.
   You’ll need to provide the same values that the developer used in the SMS push API for the SMS service provider:
   ○ Login – the login to use for the SAP SMS 365 service provider. SAP Cloud Platform mobile service for development and operations uses the login to send SMS text messages through the service provider.
   ○ Password – the encoded password associated with the login.
   ○ Originating Address – the short code or service number dedicated to a content provider account for a mobile-originating message.
4. Click Save.

1.4.3.1.3.10.2 Predefined Push Configuration

(SAP Content to Go and Fiori Client) Enable or disable preconfigured push settings.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)

3. Under Predefined Push Configuration, enable or disable preconfigured push for the application.
   - Enable Push for SAP Content to Go
   - Enable Push for SAP Fiori Client

   **Note**
   Predefined push is only supported for the app store version of SAP Fiori Client for iOS and Android.

4. Click Save.

### 1.4.3.1.3.10.3 Custom Push Provider

(Optional) Enable the custom push provider to manage push notifications for the application at the device platform level, rather than for all device platforms.

**Prerequisites**

Create the application on SAP Cloud Platform mobile service for development and operations.

**Context**

Enabling custom push providers gives you more control for managing push for the application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)
3. Under Custom Push Provider, select one or more device platform types.
4. Configure custom push provider settings for the selected device platforms. Entries are required unless stated otherwise.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>(Optional) A display name for the application. By default, the application ID is used, for example, com.sap.today.</td>
</tr>
</tbody>
</table>
### Property Description

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>The URL of the push service that accepts push message requests from SAP Cloud Platform mobile service for development and operations. The URL can be anywhere on the Internet (use “http://” or “https://” for an Internet destination), but cannot be accessed through Cloud Connector.</td>
</tr>
<tr>
<td>User Name</td>
<td>The user name to connect to the push URL.</td>
</tr>
<tr>
<td>Password</td>
<td>The user password to connect to the push URL.</td>
</tr>
</tbody>
</table>

5. Click **Save**.

### 1.4.3.1.3.10.4 Android Push Notifications

To enable client applications to receive Google Cloud Messaging (GCM) notifications, configure Android push notifications for the selected application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications**.
2. Select an application, and then select **Push Notification** under **Assigned Features** (or add it first)
3. Under Android, enter the access key for API key. This is the access key you obtained for your Google API project ([https://developers.google.com/cloud-messaging/http-server-ref](https://developers.google.com/cloud-messaging/http-server-ref)).
4. Enter a value for Sender ID. This is the project identifier.

### 1.4.3.1.3.10.5 Apple Push Notifications

To enable client applications to receive APNS notifications, configure Apple push notifications for the selected application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications**.
2. Select an application, and then select **Push Notification** under **Assigned Features** (or add it first)
3. If you do not want to receive APNS push notification, under Apple, select **None**.
4. To configure APNS for a development and testing environment, select **Sandbox**; to configure APNS for a production environment, select **Production**.
a. Click Browse to navigate to the certificate file.
b. Select the file, and click Open.
c. Enter a valid password.
5. Save your changes.

1.4.3.1.3.10.6 BlackBerry Push Notifications

To enable client applications to receive BES/BIS notifications, configure BlackBerry push notifications for the selected application.

Prerequisites

If you intend to use push synchronization with BlackBerry devices, enable push synchronization in the BlackBerry server, using the BlackBerry server documentation.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)
3. Under BlackBerry, select the push type.
   ○ Select None if you do not want to configure Blackberry push notification.
   ○ Select BES to configure Blackberry Enterprise Server (BES) native notification properties.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server URL</td>
<td>Address in the form http://&lt;domain_name&gt; or &lt;IP_address&gt;:&lt;port_Number&gt;/pap.</td>
</tr>
<tr>
<td>User Name</td>
<td>(Optional) User who is accessing the URL.</td>
</tr>
<tr>
<td>Password</td>
<td>User password to connect to the URL. If you set a user name, you are required to also enter a password.</td>
</tr>
</tbody>
</table>

○ Select BIS to configure Blackberry Internet Server (BIS).

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Server URL</td>
<td>Address in the form <a href="https://cp">https://cp</a>&lt;XXXX&gt;.pushapi.eval.blackberry.com/mss/PD_&lt;pushRequest&gt;</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Listener Port</td>
<td>The push listener port for BIS notifications</td>
</tr>
<tr>
<td>Application ID</td>
<td>The unique identifier assigned to the registered push application service</td>
</tr>
<tr>
<td>Password</td>
<td>The configuration property provided by BlackBerry for BIS push.</td>
</tr>
</tbody>
</table>

### 1.4.3.1.3.10.7 Windows Push Notifications

To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

#### Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications**.
2. Select an application, and then select **Push Notification** under **Assigned Features** (or add it first).
3. Under **WNS**, enter the application credentials, which are provided by the application developer.

#### Table 29:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package SID</td>
<td>Package security identifier</td>
</tr>
<tr>
<td>Client Secret</td>
<td>Client secret information</td>
</tr>
</tbody>
</table>

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

### 1.4.3.1.3.10.8 Windows Phone Push Notifications

To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows phone users running mobile applications, configure Microsoft push notification services (MPNS) for the selected application.

#### Context

---

**i Note**

Only unauthenticated push notification is supported; authenticated push notification for MPNS is not supported.
**Procedure**

1. In Mobile Service for Development and Operations cockpit, select *Mobile Applications*.
2. Select an application, and then select *Push Notification* under *Assigned Features* (or add it first).
3. Under *MPNS*, select *Enable MPNS HTTP Push* to send HTTP push notifications to the device.
4. (Optional) Configure push notifications for each device type supported.

**1.4.3.1.3.11 Using the Sample Back End**

A sample OData service is available for developers to use during development and testing. The sample OData service also supports delta tokens, which let you evaluate how delta tokens are handled in your test application.

**Prerequisites**

- The Developer role.
- For the Developer role, the root service URL is the subscribed JAVA application URL followed by `<subscribed_java_app_url>/SampleServices/ESPM.svc`.
- For the Administrator role, the root service URL is in the format `/mobileservices/Admin/ESPM.svc/` Configure one sample OData service and Developer account per tenant.

**Context**

Administrators can view the sample back-end service via the cockpit. You can view the root service and metadata URLs, and generate sample sales orders and purchase orders for multiple entity sets. You can also view the data for each entity in a separate text file, and reset the sample data.

Developers can simply add the service to the connectivity feature, to consume the service in the application. Delta tokens let you retrieve the changes that have been applied to a service. You can send a request to the back end and the delta token is appended to the response.

By default, the application uses the following settings to provide a security check, and enable the developer to access the service using `https://<mobile service host>/com.sap.edm.sampleservice`. For the SSO Mechanism, Application-to-Application SSO is used, and cannot be modified.

Once you add the service to an application, you can use the service via the cockpit. You must generate the initial sample sales order and purchase order records for each entity set. You can also view the root service and metadata URLs, view the data for each entity in a separate text file, and reset the sample data. You can generate a demo HTTP error if a quantity value of `SalesOrderItem` is below zero. Use the OData error handling API to show the error for your application.
The demo error handling of OData sample service is to add a data validation functionality for the *quantity* field of *SalesOrderItem*. When you try to create or update the quantity value using the POST operation, which is less than or equals to 0, the response is:

```json
{
   "error": {
      "code": null,
      "message": {
         "lang": "en",
         "value": "The value of quantity should be greater than zero."
      }
   }
}
```

along with status code 400

The Products entity includes sample images identified using the property "PictureUrl". The images can be viewed in the cockpit, and provide image examples for tutorials. You cannot change the image examples.

```xml
<EntityType Name="Product">
   <Key>
      <PropertyRef Name="ProductId"/>
   </Key>
   <Property Name="Category" Nullable="true" Type="Edm.String" MaxLength="40"/>
   <Property Name="CategoryName" Nullable="true" Type="Edm.String" MaxLength="40"/>
   <Property Name="CurrencyCode" Nullable="true" Type="Edm.String" MaxLength="5"/>
   <Property Name="DimensionDepth" Nullable="true" Type="Edm.Decimal" Scale="4" Precision="13"/>
   <Property Name="DimensionHeight" Nullable="true" Type="Edm.Decimal" Scale="4" Precision="13"/>
   <Property Name="DimensionUnit" Nullable="true" Type="Edm.String" MaxLength="3"/>
   <Property Name="DimensionWidth" Nullable="true" Type="Edm.Decimal" Scale="4" Precision="13"/>
   <Property Name="LongDescription" Nullable="true" Type="Edm.String" MaxLength="255"/>
   <Property Name="Name" Type="Edm.String"/>
   <Property Name="PictureUrl" Nullable="true" Type="Edm.String" MaxLength="255"/>
   <Property Name="Price" Nullable="true" Type="Edm.Decimal" Scale="3" Precision="23"/>
   <Property Name="ProductId" Nullable="false" Type="Edm.String" MaxLength="10"/>
   <Property Name="QuantityUnit" Nullable="true" Type="Edm.String" MaxLength="3"/>
   <Property Name="ShortDescription" Nullable="true" Type="Edm.String" MaxLength="255"/>
   <Property Name="SupplierId" Nullable="true" Type="Edm.String" MaxLength="10"/>
   <Property Name="UpdatedTimestamp" Type="Edm.DateTime"/>
   <Property Name="Weight" Nullable="true" Type="Edm.Decimal" Scale="3" Precision="13"/>
   <Property Name="WeightUnit" Nullable="true" Type="Edm.String" MaxLength="3"/>
   <NavigationProperty Name="StockDetails" ToRole="Stock" FromRole="Product" Relationship="ESPM.Product_Stock_One_One1"/>
   <NavigationProperty Name="SupplierDetails" ToRole="Supplier" FromRole="Product" Relationship="ESPM.Supplier_Product_One_Many0"/>
</EntityType>
```
Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** ▶ **Native/Hybrid**.
2. Select an application.
3. To use the sample OData service, in the **Info** tab, click **+** icon in the **Assigned Features** section.
4. In the Add feature window, select **Sample Back End** and click **OK**.
5. Select the **Entity Sets**.
6. Select an operation for the selected entity set.
   - To open or save a text file of the JSON file contents for the selected entity, such as `Customers.json`, click the **icon.
   - To generate ten sample sales orders (each click generates ten more), click the ** icon.
   - To generate ten sample purchase orders (each click generates ten more), click the ** icon.
   - To reset the data to an initial state of ten records, click the ** icon.
7. Select a different entity set from the list, such as Customers, Product Categories, Product Texts, Products, and so on.

You see a table of properties for the selected entity.

Table 30: Entity Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Property</td>
<td>List of entity properties, such as City, Country, and CustomerId.</td>
</tr>
<tr>
<td>Data Type</td>
<td>The property's data type, such as Edm.String, or Edm.DateTime.</td>
</tr>
<tr>
<td>Can be nulled</td>
<td>Whether the property is nullable, typically true or false.</td>
</tr>
<tr>
<td>Property Type</td>
<td>The property type, such as Simple.</td>
</tr>
</tbody>
</table>

Next Steps

Once the service is configured, the developer can access it using the following URL after onboarding:


### Note

The sample OData service automatically assigns the sample OData destination named `com.sap.edm.sampleservice` when you add a sample back-end feature to a mobile application. The sample back-end feature allows you to use the configuration user interface to maintain or configure back-end data. The `com.sap.edm.sampleservice` takes care of the runtime connectivity channel from device to OData back end.
You can preview embedded sample back-end images in Mobile Service for Development and Operations cockpit.

### 1.4.3.1.3.11.1 Viewing Sample Pictures

You can preview embedded sample back-end images in Mobile Service for Development and Operations cockpit.

#### Prerequisites

- Log into Mobile Service for Development and Operations cockpit with a Developer role.
- Include the Sample Back End in the application definition.
- You must provide the right security information to preview images.

#### Context

You can preview images in several ways:

- **Direct way:** `https://<mobile_service_host>/<PictureURL_value>`.
- If the Sample Back End feature has been assigned to an application, you can modify the PictureUrl value, and preview the image.
- If the Sample Back End feature has been assigned to an application and a registration has been generated, you can view the PictureUrl with the `X-SMP-APPCID` in the request header.

#### Procedure

1. In Mobile Service for Development and Operations cockpit, select `Mobile Applications ➔ Native/Hybrid`.
2. Select an application, and then select `Sample Back End` under `Assigned Features` (or add it first).
3. Select `Products` from `Entity Sets`, and click view data.

Open the download to see the Products data, including the Picture Url values for all products in the catalog.

```json
{
   "d": {
      "results": [
         {
            "__metadata": {
               "id": "https://<mobile_service_host>/Admin/ESPM.svc/Products('HT-1601')"
            }
         }
      ]
   }
}`
4. Use the PictureUrl value to preview the picture, using one of the available options.

5. (Optional) Access the picture directly using a URL comprised of the host name and PictureUrl: https://<mobile_service_host>/<PictureUrl_value>. For example, https://<mobile_service_host>/SampleServices/ESPM.svc/resources/images/HT-1601.jpg.

6. (Optional) If the Sample Back End feature has been assigned to an application, you can modify the PictureUrl value, and preview the image. Use the format: https://<mobile_service_host>/<root_URL>/resources/images/<picture_name>.jpg. For example, https://<mobile_service_host>/mobileservices/Admin/ESPM.svc/resources/images/HT-1601.jpg.

   Note
   Select the Configuration tab to obtain the <root_URL> value.

7. (Optional) If the Sample Back End feature has been assigned to an application and a registration has been generated, you can visit the picture url with the X-SMP-APPCID in the request header: https://<mobile_service_host>/com.sap.edm.sampleservice/resources/images/<picture_name>.jpg. For example, https://<mobile_service_host>/com.sap.edm.sampleservice/resources/images/HT-1601.jpg.

1.4.3.1.3.12 Using the Cloud Build Service

Build custom Fiori Clients for your Fiori Launchpad or individual SAP Fiori business apps, and enable them to use the SAP Mobile Platform SDK. Use the Cloud Build service to build the binary.

Prerequisites

- Create the Fiori Destination URL in Destinations, or identify a custom URL. The Fiori Destination can point to a single application, or to the Fiori Launchpad.
- Define the Fiori application in Mobile Applications > Native/Hybrid. You can use the Hybrid application type and add the Cloud Build feature, or you can use the Fiori application type, which automatically includes the Cloud Build feature.
Identify the splash screen and application launch icons that you plan to implement. File size may not exceed 50KB.
- Splash screen: portrait mode (1536 by 2048 pixels)
- Splash screen: landscape mode (2048 by 1536 pixels)
- Application launch: 152 by 152 pixels

Create signing profiles needed for each platform type. See Managing Signing Profiles for information.

Context

You can build custom Fiori Clients for your Fiori Launchpad or individual SAP Fiori business apps, and enable them to use the SAP Mobile Platform SDK. As a result, these apps can be programmed and configured to interact with native device features, push notifications and other mobile services features.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid.
2. Select an application, and then select Cloud Build under Assigned Features (or add it first).
3. On the Configuration tab, configure packaging details.
   a. Under Details, provide the Cloud Build details.

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Device App Name</td>
<td>The name used to identify the device application, such as FioriAppWeather</td>
</tr>
<tr>
<td>Fiori Destination Name</td>
<td>Select a Fiori destination name, or select Custom URL and enter the URL.</td>
</tr>
<tr>
<td>Fiori Destination URL</td>
<td>If you selected an existing Fiori destination name, the Fiori destination URL cannot be edited.</td>
</tr>
</tbody>
</table>

   b. Under Multimedia, upload splash screen and launch icons in the sizes specified.
   c. Click Save to save the configuration.
4. Select the Build History tab. You can view the build history for each build package; view and download build logs; download binaries; and modify build settings to create a new build package. See Modifying Cloud Build Settings for details.
Table 32: Cloud Build History

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Current state of the build, such as Building, Build Failed, and Build Complete.</td>
</tr>
<tr>
<td>Debug Enabled</td>
<td>Whether the debugging tool is enabled.</td>
</tr>
<tr>
<td>Operating System</td>
<td>The operating system of the build, such as iOS or Android.</td>
</tr>
<tr>
<td>Form Factor</td>
<td>The device type, such as tablet or phone.</td>
</tr>
<tr>
<td>App Version</td>
<td>The application version.</td>
</tr>
<tr>
<td>Built On</td>
<td>The date of the build, in the format YYYY-MM-DDT HH:MM SS SubS.</td>
</tr>
<tr>
<td>Certificate Expiration</td>
<td>The date the certificate expires.</td>
</tr>
<tr>
<td>Actions</td>
<td>Actions you can take, such as Delete, to delete the build.</td>
</tr>
</tbody>
</table>

5. (Optional) Select the Info tab to view feature details.

1.4.3.1.3.13 Defining Application Security

Define the settings that control user authentication behavior for the selected application.

Prerequisites

If you define an application that uses Basic authentication, the default behavior is to delegate Basic authentication to an SAP ID service (“Default Identity Provider”).

Alternatively, you can configure the application to use the global System for Cross-Domain Identity Management (SCIM) settings, or you can override the global settings. To override the default, you must know the SCIM URL and the proxy type (Internet or On Premise).

To check the current global SCIM settings, select Settings ➤ Account Security in the cockpit. You can modify the SCIM settings later.

- If Basic authentication is to be delegated to an on-premise user store, you must:
  - Set up the on-premise user store. See Using an On-Premise User Store for details on the different ways to set up the on-premise user store: https://help.hana.ondemand.com/help/frameset.htm?04cbd0f30d524612aa438ed0b0eed217.html.
  - Update the SCIM configuration to use “HCP SCIM”. See Configuring SCIM Authentication for details about configuring SCIM using Settings ➤ Account Security.
- If Basic authentication is to be performed against a URL that supports HTTP basic authentication, you must:
- Update the SCIM configuration to use "Mobile Service SCIM". See Configuring SCIM Authentication for details about configuring SCIM using \( \text{Settings} \rightarrow \text{Account Security} \).
- Specify that URL in the account security settings under Default Authentication Destination.

**Note**

Additionally, you can override the configured URL in the application configuration if you desire to delegate the basic authentication for a particular application to a different URL (but only if the global SCIM configuration in account security is configured to use "Mobile Service SCIM").

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select \( \text{Mobile Applications} \rightarrow \text{Native/Hybrid} \).
2. Select an application, and then select \( \text{Security} \) under \( \text{Assigned Features} \).
3. (Optional) Select \( \text{Info} \) to view feature details.
4. Select \( \text{Configuration} \) to configure application security.
5. Configure Application Settings:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSRF Protection</td>
<td>Enable protection against cross-site request forgery (CSRF) attacks for the selected application. This option protects all services, such as registration, with CSRF tokens. Proxy endpoints are not protected, since they may be protected on the back end.</td>
</tr>
<tr>
<td>Ignore Case for User Name</td>
<td>This is useful if the identity provider, such as SCIM or SAP ID service, is case-sensitive, and when push notifications are processed for SAP Fiori applications that expect user names to be matched case-insensitively, as in ABAP systems. By default, SAP Cloud Platform mobile service for development and operations compares names using case-sensitive matching, to ensure that application connections are not shared by multiple users, and for push notifications.</td>
</tr>
</tbody>
</table>
### Security Configuration

Select the appropriate security configuration:

- **None** – anonymous authentication. No authentication challenge is sent; requests are processed anonymously. This typically should not be used for business applications.
- **Basic** – HTTP-Basic (user name and password) authentication. Review the SCIM configuration under [Settings > Account Security](#) to verify that the desired target is configured to delegate the basic authentication. You can override the HTTP URL to delegate application user authentication to a different URL for a specific application, but only if Mobile Service SCIM is set in [Account Security](#).
- **Certificate** – X.509 certificate authentication.
- **SAML** – (default) SAML-based SSO authentication.
- **OAuth** – token-based authentication.

See [User Identity Propagation Methods](#) and [Application Authentication](#) for details about security configurations.

### Require One Time Passcode

(Optional) If the value of Security Configuration is Basic, Certificate, or SAML, select this option to require two-factor authentication for the application. When you run the application, you are prompted for a one-time password.

6. **Configure application threshold settings:**

Table 34: Threshold Settings

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Registration Threshold</strong></td>
<td>The number of items allowed per second. Leave blank (default) or set to 0 to remove threshold. Set a threshold value from 1 - 2147483647. If the number of incoming registration requests per second exceeds the value, requests are throttled and the server generates HTTP error code “429 TOO MANY REQUESTS”.</td>
</tr>
<tr>
<td><strong>Online Request Threshold</strong></td>
<td>The number of items allowed per second. Leave blank (default) or set to 0 to remove threshold. Set a threshold value from 1 - 2147483647. If the number of incoming online requests exceeds the value, requests are throttled briefly and the server generates HTTP error code “429 TOO MANY REQUESTS”.</td>
</tr>
<tr>
<td><strong>Offline Request Threshold</strong></td>
<td>The maximum number of requests allowed per second. Leave blank (default) or set to 0 to remove threshold. Set a threshold value from 1 - 2147483647. If the number of incoming offline requests exceeds the value, the offline requests are left in the request queue and if the offline request in the server exceed 8 minutes, then the offline request is discarded from the server. Another offline request is automatically sent to the server by the offline SDK to retry again, and this offline request can either succeed, or fail, or hang, depending on the server conditions. So, you might see some unexpected delay due to the throttling and retries.</td>
</tr>
</tbody>
</table>

7. Click **Save**.
1.4.3.2 Configuring SAP Content to Go Applications

You can configure the predefined SAP Content to Go application from the cockpit for the development and operations mobile service.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications SAP Content to Go.
2. (Optional) View the information, APIs, user registration, and usage analytics details for the selected SAP Content to Go application.
   - The Information tab is a summary of settings and application connections.
   - Select APIs tab to view a list of frequently used API URLs.
     - Server API – is the root API of mobile service for development and operations. For example, you will use this API to integrate from a third-party website.
     - Registration API – you can send this URL to users to help them register their SAP Content to Go applications by providing them with the server URL and the OAuth client ID.
   - Select User Registration tab to view details about valid application registration and users.
   - Select Usage Analytics tab to see usage statistics for the application:
     - Select the Time Frame filter for the registration time period, including Today, Yesterday, Last 7 Days, Last 4 Weeks, Last 3 Months, and Last 12 Months.
     - Indicate whether to show data in bar-chart format (column or line), or table-chart format.
4. Select Connectivity [page 58] feature to define destinations for the application. You can either create a new destination or use an existing one.

i Note
SAP Content to Go application destinations must use No Rewriting mode.
5. Select **Push Notification** [page 79] feature to configure predefined push, which enables you to use the push settings that are already preconfigured in the app that was downloaded from the Apple App Store, and also define categories for card types.

6. Select **Security** [page 189] feature to configure the OAuth access tokens and refresh token refresh lifetime.

See the **Configuring Applications** topics for details, and **Save** your changes.

7. Select **Upload Theme** to define card types for Fiori cards.

**Uploading Themes for Fiori Cards** [page 96]

You can upload themes to change the appearance of Fiori cards. For example, you can change background colors or font types.

**Related Information**

*SAP Content to Go Overview* [page 40]

*Manage SAP Content to Go Card Types* [page 144]

### 1.4.3.2.1 Uploading Themes for Fiori Cards

You can upload themes to change the appearance of Fiori cards. For example, you can change background colors or font types.

**Context**

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** ➤ **SAP Content to Go** ➤
2. Select **Upload Theme**.
3. Enter the properties.
Table 35: Theme Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Card Class</td>
<td>Select the class for your Fiori cards.</td>
</tr>
<tr>
<td></td>
<td>○ Object Page — represents business objects using a standardized template based on Fiori elements, for example, a sales order.</td>
</tr>
<tr>
<td></td>
<td>○ Overview Page — also based on Fiori elements, but overview pages are typically graphical-based cards used for analytical content. For more information about SAP Fiori elements, see Introduction to SAP Fiori Elements.</td>
</tr>
<tr>
<td>Theme Name</td>
<td>Select one of the predefined themes.</td>
</tr>
<tr>
<td>Theme to Upload</td>
<td>Browse for the zip file containing the theme. For information about creating themes, see UI Theme Designer.</td>
</tr>
</tbody>
</table>

4. Select Upload to upload your theme.

**Note**

If you select a predefined SAP theme, you must first select Activate SAP Theme to upload your theme.

### 1.4.3.3 Enabling Applications to Discover Configurations

Using Mobile Service for Development and Operations cockpit, you can publish application configurations to the SAP Discovery Service, on which mobile applications can find their connection settings. You can update or delete published configurations at any time.

**Prerequisites**

The system administrator requests that the service provider for SAP Cloud Platform mobile service for app and device management builds discovery capabilities into cloud applications, eliminating the need to manually configure applications, and speeding application adoption. You receive an e-mail with the application service provider account details.

1. Adding a Discovery Service Provider [page 98]

   Add a Discovery Service provider to your account. This enables you to use SAP Cloud Platform mobile service for app and device management. If you want to replace an existing service provider, you must delete the existing service provider first.

2. Adding Service Provider Domains [page 99]

   Add a domain for the application service provider.

3. Sharing the Domain [page 100]

   Share a domain with other application service providers.

4. Publishing Application Configuration [page 100]

   Publish an application configuration on SAP Mobile Place. You can publish the default application configuration, or customize it by adding name:value pairs or a JSON construct.
1.4.3.3.1 Adding a Discovery Service Provider

Add a Discovery Service provider to your account. This enables you to use SAP Cloud Platform mobile service for app and device management. If you want to replace an existing service provider, you must delete the existing service provider first.

Prerequisites

To use the Application Configuration Discovery Service:

- Request this service by creating an incident using the Service Market Place for component MOB-SEC-ASP.
- Enter the customer ID, Cloud Platform account name, S-User ID, and a technical contact e-mail address in the ticket.

**Note**
The information required to add a service provider to your cloud platform account is sent to the e-mail address listed in the ticket.

Context

Keep in mind these guidelines when working with discovery service providers:

- A provider only can be used once. If you delete an existing provider, you cannot use it again.
- If you delete a provider, all its associated domains (both shared and unshared) are also deleted.
- You must request a new provider if you plan to replace an existing one.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Discovery Service.
2. Click Add Provider.
   
   Verify that you have the system generated e-mail with the provider ID and activation token open, so you can copy and paste the values into the dialog.
3. Enter the Provider ID and Activation Key, and click Save. After you add a service provider, the Delete Provider button is enabled, and the Add Provider button is disabled.
   
   You can add domains to this provider.
4. (Optional) To remove a provider, click Delete Provider, and confirm.
   
   Once you click OK, the Add Provider button is enabled, and the Delete Provider button is disabled.
i Note
Keep in mind the guidelines listed above if you plan to delete a discovery service provider.

Task overview: Enabling Applications to Discover Configurations [page 97]

Next task: Adding Service Provider Domains [page 99]

Related Information

Publishing Application Configuration [page 100]

1.4.3.3.2 Adding Service Provider Domains

Add a domain for the application service provider.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Discovery Service.
2. Click the plus sign.
3. Enter the Domain.
4. Click Save.
5. (Optional) To delete a domain, select Delete.

Task overview: Enabling Applications to Discover Configurations [page 97]

Previous task: Adding a Discovery Service Provider [page 98]

Next task: Sharing the Domain [page 100]
1.4.3.3.3 Sharing the Domain

Share a domain with other application service providers.

Prerequisites

To share a domain, the current provider must own the domain.

Note

Each domain has a unique owner.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Discovery Service.
2. Select a domain, then select Share.
3. Enter the Provider ID with which you want to share the domain, and click Save.
   The domain appears as a shared provider.
4. (Optional) To stop sharing domains with other providers:
   a. Under Ownership, click the link for the domain you want to stop sharing.
   b. Select the providers to stop sharing, and click Unshare.

Task overview: Enabling Applications to Discover Configurations [page 97]

Previous task: Adding Service Provider Domains [page 99]

Next task: Publishing Application Configuration [page 100]

1.4.3.3.4 Publishing Application Configuration

Publish an application configuration on SAP Mobile Place. You can publish the default application configuration, or customize it by adding name:value pairs or a JSON construct.

Prerequisites

You must add a discovery service provider before you can publish an application configuration to SAP Mobile Place. See Adding a Discover Service Provider.
Context

You can also publish application configurations for the Content to Go and Fiori Client applications.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications.

2. Choose an application, and select Publish.

   **Note**
   The default Publish will fail with the message Application not published to Discovery Service if no discovery service provider has been added. See Adding a Discover Service Provider.

3. In Publish Configuration Information for Discovery Service, choose a configuration option.

<table>
<thead>
<tr>
<th>Configuration Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Default</td>
<td>Publishes the default application configuration to the discovery service.</td>
</tr>
<tr>
<td>Custom Properties</td>
<td>Add one or more key value pairs to customize the application configuration that is published to the discovery service.</td>
</tr>
<tr>
<td>Custom JSON</td>
<td>Add a JSON construct to customize the application configuration that is published to the discovery service.</td>
</tr>
</tbody>
</table>

4. (Default) Select Default to use the default application configuration. A confirmation message indicates the application configuration was published successfully to the discovery service.

5. (Optional) Select Custom Properties to add key value pairs to the application configuration.
   a. Click + to add a key value pair.
   b. In Key, add the key for the pair.
   c. In Value, add the value for the pair.
   d. Click + to add another key value pair; or click - to remove one, and confirm.
   e. Click Save. The key value format is validated, and the application configuration is published to the discovery service.

   If you receive an error, click OK. Resolve the formatting problem to continue.

6. (Optional) Select Custom JSON to customize the generated JSON code.
   a. In the text box, provide the JSON string, for example: {"key1":"value"}. Use the format:

   ```json
   {"key1":"value1", "p2":2, "k3":{"k3p1":"v1", "k3p2":2}}
   ```
   
   b. Click Save. The application configuration is published to the discovery service.

7. (Optional) From Applications, choose an application, and select Unpublish to remove a configuration from SAP Mobile Place.
The application configuration is removed from SAP Mobile Place, and Configuration Published is set to No on the application Overview page.

Task overview: Enabling Applications to Discover Configurations [page 97]

Previous task: Sharing the Domain [page 100]

1.4.3.4 Configuring Agentry Applications

Create an application definition that enables you to manage an Agentry application using the cockpit. The application definition includes Name, Application ID, Vendor and the Creation Date.

Procedure

1. In the cockpit, select Mobile Applications Agentry Applications, and open an Agentry application.
2. (Optional) View information about the selected Agentry application.
   - The Info tab is a summary of application details and assigned features.
     - Select Edit in the Application Details section to edit existing application details.
     - In the Assigned Features section, click Agentry Applications to configure various features:
       - For application-specific settings, see Defining Agentry Application Settings [page 106].
       - For connectivity, see Defining Back-end Connections for Agentry [page 64].
       - To configure predefined push, which lets you use the preconfigured push settings, select Push Notifications [page 79].
       - (Optional) View Assigned Feature details for Agentry Applications in the Info tab.
   - Select Users tab to see information about application users.
     - (Optional) To reorder user information, use the column sorting options.
     - (Optional) In Filter By, enter a value to search for a user name, unique ID, device type or last connection time frame.

You see the following information about users.

Table 36: Field Descriptions

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Name of the person using the application.</td>
</tr>
<tr>
<td>Unique ID</td>
<td>User ID.</td>
</tr>
<tr>
<td>Application</td>
<td>Application ID.</td>
</tr>
<tr>
<td>Device Type</td>
<td>The parameter value, such as Android or iPhone, sent by the device during connection. “Unknown” indicates the device type cannot be detected.</td>
</tr>
</tbody>
</table>
Field | Value
--- | ---
Last Connection Time | The date and time of the last time the user connected to SAP Cloud Platform mobile service for development and operations, in the format MMM DD, YYYY HH:MM:SS AM/PM. You can also disconnect one or more Agentry user connections.

Defining Agentry Applications [page 103]
Use the cockpit to create and manage a new Agentry application.

Importing an Agentry Application [page 104]
Import an Agentry application from your local machine.

Restarting an Agentry Application [page 105]
Restart an Agentry application.

Exporting an Agentry Application [page 105]
Export an Agentry application to your local machine.

Deleting an Agentry Application [page 105]
Delete an Agentry application.

1.4.3.4.1 Defining Agentry Applications

Use the cockpit to create and manage a new Agentry application.

Procedure

1. In the cockpit, select **Mobile Applications > Agentry Applications > New**
2. Enter the application properties.
Table 37: Application Properties

<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 is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Cloud Platform mobile service for development and operations, and client applications use the Application ID when sending requests to the server. An application ID:  
  ○ Must be unique  
  ○ Must start with an alphabetic character  
  ○ Can contain only alphanumeric characters, underscores, and periods  
  ○ Can contain up to 64 characters  
  ○ Cannot include spaces  
  ○ Cannot begin with a period, and cannot contain two consecutive periods  
  ○ Cannot be any of these case-sensitive keywords: Admin, AdminData, Push, amp_cloud, resource, test-resources, resources, Scheduler, odata, applications, Connections.public, lcm  
  We recommend that you assign IDs that contain a minimum of two periods, for example, com.sap.mobile.app1. |
| Name    | The application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.   |
| Description | (Optional) The description can contain up to 255 alphanumeric and special characters                                                                 |
| Vendor  | (Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long. |

3. Click Save.

1.4.3.4.2 Importing an Agentry Application

Import an Agentry application from your local machine.

Procedure

1. In the cockpit, select Mobile Applications Agentry Applications, and click Import.
2. Browse to and open the Agentry application file you want to import.
3. Click Save.
1.4.3.4.3 Restarting an Agentry Application

Restart an Agentry application.

**Procedure**

1. In the cockpit, select [Mobile Applications] [Agentry Applications].
2. Identify an Agentry application, and open it.
3. Click **Restart**.

1.4.3.4.4 Exporting an Agentry Application

Export an Agentry application to your local machine.

**Procedure**

1. In the cockpit, select [Mobile Applications] [Agentry Applications].
2. Identify the Agentry application you want to export, and open it.
3. Click **Export**.

1.4.3.4.5 Deleting an Agentry Application

Delete an Agentry application.

**Procedure**

1. In the cockpit, select [Mobile Applications] [Agentry Applications].
2. Identify the Agentry application you want to delete, and open it.
3. Click **Delete**, then **OK** to confirm.

**Note**

Once an application has been deleted, users cannot use it. All existing logs and traces are deleted and cannot be retrieved.
1.4.3.5 Defining Agentry Application Settings

(Optional, applies only to Agentry) Configure application-specific settings for the selected application, using mobile service for development and operations Cockpit, configuration files, or other tools.

1.4.3.5.1 Publishing Agentry Apps

To publish a new Agentry application, create or upload the Agentry application ZIP file to SAP Cloud Platform mobile service for development and operations. To update an Agentry application, you must publish a new version to change the definitions.

Prerequisites

If you are using an Adaptive Server Enterprise database, before you publish in a cluster environment, verify that the database procedure cache size is larger than the Agentry application size. If it is not, publishing is successful on the server node, but inserting into the database and attempting to push to other server nodes in the cluster fails with the message Encountered error while saving the application to the database. If you proceed without updating the cache size, old and new versions of the application will be used by different sever nodes in the cluster.

Note

The procedure to update the Adaptive Server Enterprise database cache size may vary by version, but see the following for a good example: Procedure Cache.

Context

The application developer can use Agentry Editor to create an application ZIP file, which contains production definitions, and any other files the developer included. Only production applications can be published to Mobile Service for Development and Operations cockpit.

The administrator uploads the ZIP file to Mobile Service for Development and Operations cockpit, and can configure additional Agentry settings, and change some Agentry configuration settings made by the developer.

Procedure

1. In Mobile Service for Development and Operations cockpit, select [Mobile Applications] and open an Agentry application.
2. Under Assigned Features, click **Agentry Applications**.

   When you publish a new version of an Agentry application, it appears under **Published Versions** in the Application-Specific Settings tab.

3. Configure additional Agentry settings as needed. Changes to some configuration values may require you to restart the application for the changes to take effect.

4. Click the **Publish Agentry application** button to publish the app.

### 1.4.3.5.2 Managing Agentry App Versions

View multiple versions of published Agentry applications that are currently in use, and delete versions that are no longer needed.

#### Context

When you publish a new version of an Agentry application, it appears in the **Published Versions** section of the **App Specific Settings** screen in Mobile Service for Development and Operations cockpit. Although you are allowed to maintain multiple versions of an application, it is a good practice to delete old versions to clean up the database, release memory, and improve load time performance. Before you delete a version, ensure that all users have upgraded to the newest version. Deleting active versions renders clients using those versions inoperable, and requires the client application to be reset.

#### Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications > Agentry Applications** and open an Agentry application.

2. Under Assigned Features, click **Agentry Applications**.

3. Under **Published Versions** in the Application-Specific Settings tab, review the list of current application versions.

4. Select an application version, and click the Delete icon. The application version is removed from the list.
1.4.3.5.3 Configuring Agentry Settings

After you import an Agentry application, you can change the Agentry production application definitions made by the developer, and configure additional Agentry application settings. Some settings apply to the current application, and others apply to all applications.

Context

If you change settings in a section marked "Affects All Agentry Applications", keep in mind that all Agentry applications are affected.

**Note**
The sections that appear and their order vary, depending on how the Agentry application has been developed. The settings that affect all applications are grouped at the end. The Publish and Logging sections are documented separately.

Procedure

1. In Mobile Service for Development and Operations cockpit, select [Mobile Applications > Agentry Applications] and open an Agentry application.
2. Under Assigned Features, click [Agentry Applications].
3. In the Application-Specific Settings tab, under [Time Zone Aliases], map time zone names on remote systems (client devices and back-end systems) to time zone names on this server. Create a list of key-value pairs, one key-value per line, in the format: "key=value". The keys are the time zone names on the remote systems; the values are the equivalent time zone names for the server's host operating system. For example, enter **Central Standard Time=America/Chicago** or **UTC+2:00=E. Europe**.

**Note**
The UTC notation (such as UTC+4:00=Moscow Standard Time) is recommended, especially outside of the United States.

4. Under **SpinDoc**, edit the values that are exposed to the Agentry application as variables [Server Data Markup Language (SDML)], and the "Face Path" setting.

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Face Path</td>
<td><code>facePath=sql</code> `custom;sql`</td>
<td>The Face Path setting defines the search path used by the SQL back end to locate the query files, which are used for database initialization, authentication, password changes, and so forth; and the SQL script files, which are used for SQL.</td>
</tr>
</tbody>
</table>
5. Under *Configuration*, edit settings for the Agent directory and path location for configuration-related files.

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Globals File</td>
<td>Globals.ini</td>
<td>Name and location of the globals localization file. You can change this setting if the name of the file is different from the default.</td>
</tr>
<tr>
<td>Application Strings File</td>
<td>ApplicationText.ini</td>
<td>Name and location of the application strings localization file. You can change this setting if the name of the file is different from the default.</td>
</tr>
<tr>
<td>Client String Names File</td>
<td>ClientStringNames.ini</td>
<td>Name and location of the client string names localization file. You can change this setting if the name of the file is different from the default. Changes to this property require an application restart.</td>
</tr>
<tr>
<td>Client Strings File</td>
<td>ClientText.ini</td>
<td>Name of the client strings localization file. You can change this setting if the name of the file is different from the default.</td>
</tr>
<tr>
<td>Development Server</td>
<td>N/A</td>
<td>Read-only. Whether the application is development or production.</td>
</tr>
<tr>
<td>Enable Failed Transaction Logging</td>
<td>True</td>
<td>Whether to enable the failed transaction queue, which is a part of the transaction failure handling functionality. By default, when transaction failure handling is enabled, transactions that fail with a fatal error code result in the creation of a failed transaction file on the server containing the data from that transaction. If this option is set to false, the file is not generated. All remaining transaction failure handling behaviors are carried out, including the removal of the failed transaction from the client.</td>
</tr>
</tbody>
</table>

*Note*

Setting this option to false may result in lost data for fatal errors that are related to transaction processing. This option has no effect if transaction failure handling is disabled (that is, enableTransactionFailureHandling is false.)
<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Enable Override File</td>
<td>Enables.ini</td>
<td>Name and location of the enables localization file. Change this setting only if the file name is different from the default.</td>
</tr>
<tr>
<td>Enable Transaction Failure Handling</td>
<td>False</td>
<td>Whether to enable the transaction failure handling functionality within the Agentry platform. This option must be true for all aspects of this functionality to be enabled.</td>
</tr>
<tr>
<td>Failed Transaction Filename Format</td>
<td>Server-generated XML file name</td>
<td>Format of the file name for each failed transaction queue file generated by the server. The format strings <code>{userid}</code>, <code>{transaction_name}</code>, <code>{date}</code>, <code>{time}</code>, and <code>{count}</code> may be used. The file generated is always in XML format, and the file extension provided (.xml) should reflect this content type.</td>
</tr>
<tr>
<td>Failed Transaction Queue</td>
<td>FailedTransactionsQueue</td>
<td>Location of the failed transaction queue files. This path is relative to the installation folder of the server.</td>
</tr>
<tr>
<td>Images Path</td>
<td><code>&lt;Application</code></td>
<td>Location of image files in either the development or production environment.</td>
</tr>
<tr>
<td></td>
<td>`&lt;Development_or_Produ</td>
<td>ction&gt;`</td>
</tr>
<tr>
<td></td>
<td><code>&lt;Images&gt;</code></td>
<td></td>
</tr>
<tr>
<td>Localization Path</td>
<td>localizations</td>
<td>Location of localization files for use in the application. This path is relative to the server’s installation folder. If the default folder “localizations” does not exist, it must be created.</td>
</tr>
<tr>
<td>Localizations</td>
<td>None</td>
<td>One or more local names, according to the ISO 639-1 standard. This option lists the supported languages, which also require corresponding override localization files, for an application. Use semicolons to separate names.</td>
</tr>
<tr>
<td>Scripts Path</td>
<td>Application <code>Development\Scripts</code></td>
<td>If you use an Server Data Markup Language (SDML) tag in the Agentry application that includes content from a file using a relative path, this is the directory under which the file in question needs to be located.</td>
</tr>
<tr>
<td>Spin Doc INI File</td>
<td>None</td>
<td>If this value is specified, then the [SpinDoc] configuration is loaded from the named .ini file instead of from Agentry.ini.</td>
</tr>
<tr>
<td>Transmit Configuration File</td>
<td>TransmitConfigations.ini</td>
<td>Name and location of the transmit configurations override file. Change this setting if the file name is different from the default.</td>
</tr>
<tr>
<td>Trusted Certificate Store</td>
<td>None</td>
<td>Location of the trusted certificate store for the server. This store is used only when</td>
</tr>
</tbody>
</table>

SAP Cloud Platform Mobile Service for Development and Operations

SAP Cloud Platform Mobile Service for Development and Operations
clients are required to provide authentication certificates during synchronization.

The Agentry server transmit URL, built from the application ID (specified while defining the application in the cockpit), and values provided by SAP Cloud Platform. Read-only.

6. Under **Logging (Affects All Agentry Applications)**, set these logging values:

<table>
<thead>
<tr>
<th>Definition</th>
<th>Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Event Log File</td>
<td>Valid file name, or path and file name; the default is events.log.</td>
<td>The file to which event log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.</td>
</tr>
<tr>
<td>Maximum Log Size</td>
<td>Numeric value, in bytes; the default is 0.</td>
<td>The maximum size of the log file in bytes. A value of 0 indicates no maximum size. Any other value results in the log file rolling over when the file exceeds the value set here.</td>
</tr>
<tr>
<td>Log Roll Time</td>
<td>Time of day in 24-hour clock format (HH:MM); the default is 1:00am.</td>
<td>The time at which to roll log files. The server must be running for the files to roll over at the specified time. Requires server restart.</td>
</tr>
<tr>
<td>Log Rolls At Time</td>
<td>Enabled or disabled; the default is enabled.</td>
<td>Whether to roll the log files based on the time defined in Log Roll Time. Requires server restart.</td>
</tr>
<tr>
<td>Message Log File</td>
<td>Valid file name, or path and file name; the default is messages.log.</td>
<td>The file to which message log items are written. You can change the default to specify a different file name or location. The path can be relative to the server installation folder; or a full path that includes the drive letter. Requires server restart.</td>
</tr>
</tbody>
</table>

7. Under **Agentry (Affects All Agentry Applications)**, edit Agentry application settings.

<table>
<thead>
<tr>
<th>Property</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keep Alive Delta</td>
<td>3600 (seconds)</td>
<td>Interval during which the server writes something to the various log files, if nothing else has been written during that time period to indicate that the server is not dead. Changes to this property require an application restart.</td>
</tr>
<tr>
<td>Shut Down Wait</td>
<td>15000 (milliseconds)</td>
<td>Length of time Agentry waits for all of its threads to exit while shutting down, before it stops them.</td>
</tr>
<tr>
<td>System Name</td>
<td>Agentry</td>
<td>Name of the Agentry server, which appears in the title bar of the server’s GUI interface on the console. Changes to this property require an application restart.</td>
</tr>
</tbody>
</table>

8. Under **Server (Affects All Agentry Applications)**, edit administrator contact information and client timeout and notification settings.
<table>
<thead>
<tr>
<th>Properties</th>
<th>Defaults</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Administrator Email</td>
<td><a href="mailto:admin@yourcompany.com">admin@yourcompany.com</a></td>
<td>Administrator e-mail address.</td>
</tr>
<tr>
<td>Administrator Name</td>
<td>Your Name Here</td>
<td>Administrator name.</td>
</tr>
<tr>
<td>Administrator Phone</td>
<td>111-111-1111</td>
<td>Telephone number for contacting the administrator.</td>
</tr>
<tr>
<td>Allow Relogin</td>
<td>Enabled</td>
<td>Whether to allow users to relogin.</td>
</tr>
<tr>
<td>Inactive Timeout</td>
<td>7200 (seconds)</td>
<td>Length of time that an idle user can remain connected to the server before the connection is dropped.</td>
</tr>
<tr>
<td>Notify User on Logout</td>
<td>Disabled</td>
<td>Whether to notify the user on log out.</td>
</tr>
<tr>
<td>Notify User on Shutdown</td>
<td>Disabled</td>
<td>Whether to notify the user on shut down.</td>
</tr>
</tbody>
</table>

9. Restart the application if you modified any values that require it.

### 1.4.3.5.4 Updating the Agentry Configuration

Update Agentry configuration files, and deploy the changes to the SAP Cloud Platform mobile service for development and operations production environment. Agentry configuration files include back-end connection files, and override files (localization files, query files, and constants files). Configuration files may apply to the landscape, or to a particular application. In some cases, you must restart the application before the changes take effect, in others you must restart SAP Cloud Platform mobile service for development and operations.

#### Context

- Developers create configuration files as part of application development. This typically involves copying and renaming a template file to use as a base. For example, use the ClientTextBase.ini template file to create ClientText.ini, then modify ClientText.ini to change the configuration settings.
- Once the configuration files are published to SAP Cloud Platform mobile service for development and operations, they are stored in the SAP Cloud Platform mobile service for development and operations production database.
- Administrators can update configuration files, or work with developers to update them. The updated files must be republished using Agentry Editor and Mobile Service for Development and Operations cockpit.

**Note**

Do not modify the configuration files directly on the server; those changes will be overwritten by files in the SAP Cloud Platform mobile service for development and operations production database.

- Some changes or updates to application configuration files require you to restart the application before the changes take effect:
  - Configuration changes that are related to a configuration file. For example, changing the content of an existing file, like SqlBE.ini, or changing the Query Init File of a SQL back end from SqlBE.ini to SqlBE_Oracle.ini.
- Code changes. For example, changing Java code for the Java back end.
- SQL query file changes, if the SQL back end’s preloadQueries setting is true.
- Settings where Mobile Service for Development and Operations cockpit indicates it is necessary.

- Changes made in sections labeled “Affects All Agentry Applications” in the Application-Specific Settings screen require a SAP Cloud Platform mobile service for development and operations restart before the changes take effect.

**Procedure**

1. Use any text editor to update configuration files.
2. Use Agentry Editor to publish the application. Include the updated configuration files in the publish folder structure, along with the product definitions, so they are included in the ZIP file.
3. To publish the new ZIP file in Mobile Service for Development and Operations cockpit, select Mobile Applications ➤ Agentry Applications ➤ Import, and browse to the Agentry file.
4. After import, open the application, select Agentry Applications ➤ Publish Agentry application. You can complete any additional Agentry application configuration tasks as well. When you save the ZIP file, it is uploaded through the browser to SAP Cloud Platform mobile service for development and operations, and published. You are notified of success or failure.
5. Restart the application, or SAP Cloud Platform mobile service for development and operations as required. In a cluster environment, restart the application on each server in the cluster, to distribute the configuration changes. Application users can then obtain the updated application version on their devices, from the server.

**1.4.3.5.5 Configuring the Transmit Configuration File for Agentry Applications**

To override transmit values for an application, you can create a transmit configuration file. This can be useful when you are running multiple Agentry applications on SAP Cloud Platform mobile service for development and operations.

**Production Workflow**

1. (Administrator) Work with developers to create a text file called TransmitConfigurations.ini in the application project.
2. (Developer) In the text file, for each transmit configuration to override, create a section called [TransmitConfigurationName]. Place any override attributes in each section as is appropriate. Include the file when you publish the project to production. See Publishing to Production in Developer > Agentry App Development.
3. (Administrator) Import the project ZIP file to the server.
4. (Administrator) In Mobile Service for Development and Operations cockpit, select the application and navigate to Application-Specific Settings. Under Configuration, make sure the value of Transmit Configuration File points to this ZIP file location.

5. (Administrator) To initiate the changes, restart the application on each server.

**Note**

- Make changes to the `TransmitConfigurations.ini` file in the development environment, not on SAP Cloud Platform mobile service for development and operations, and publish a new ZIP file in Mobile Service for Development and Operations cockpit. Do not modify the file on the server, or it will be overwritten when you upgrade the server.
- Restart each server in a cluster to distribute the changes.

**TransmitConfigurations.ini**

The `TransmitConfigurations.ini` file has one section for each transmit configuration that is defined in the application, where `<TransmitConfigurationName>` is the name of the definition within the application project. Each section can contain override values for any and all attributes within a given transmit configuration definition.

The following example shows several options that are valid within each section of the `TransmitConfigurations.ini` override file. The value for `stayLoggedIn` overrides the general attribute Stay Logged In.

```
[<TransmitConfigurationName>]
stayLoggedIn=true
retryPeriod=30
retryAttempts=5
onlineRetryAttempts=5
serverInactiveTimeoutOverride=300
```

All other attributes within the transmit configuration definition can be overridden using the proper key and value pair. Any attributes that are not overridden in `TransmitConfigurations.ini` are sent to Agentry Clients as they are defined in the application project.

If you have multiple Agentry applications, each server should have its own version of `TransmitConfigurations.ini` that includes overrides for the required configuration settings that are unique to that server instance.

**TransmitConfigurations.ini Attributes**

For information about definition attributes, see Developer > Agentry App Development > Agentry Language Reference > Application Level Definitions Overview > Transmit Configuration.
Table 38: General Settings

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Name</td>
<td></td>
<td>Name of the transmit configuration, which must be unique for all transmit configurations defined for the application.</td>
</tr>
<tr>
<td>displayName</td>
<td>Display Name</td>
<td>Any printable text</td>
<td>Each transmit configuration defined in the application is listed in the Transmit Dialog on the Agentry application, so choose an appropriate Display Name that is meaningful to the application end user.</td>
</tr>
<tr>
<td>connectType</td>
<td>Connect Type</td>
<td>WebSockets over HTTPS</td>
<td>Communications protocol used when synchronizing with SAP Cloud Platform mobile service for development and operations.</td>
</tr>
<tr>
<td>xmitConfigGroup</td>
<td>Group</td>
<td>Transmit Configuration Group Name</td>
<td>Group into which the transmit configuration is organized within the application. There are two default options, Fast and Slow, or you can create a new group by entering a name in this field. Any group name you enter here becomes available to all transmit configurations within the same application project.</td>
</tr>
<tr>
<td>failoverTo</td>
<td>Failover to</td>
<td>Name of other transmit configuration definition.</td>
<td>You can select a failover transmit configuration to be used if an Agentry application cannot connect to a server using the first transmit configuration. You can assign this attribute to any other transmit configuration within the application.</td>
</tr>
<tr>
<td>checkDataTables</td>
<td>Check Data Tables</td>
<td>true/false</td>
<td>Specifies whether the data tables within the application are synchronized when the transmit configuration is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: Consider not selecting this attribute for transmit configurations that are intended for slower connection types.</td>
</tr>
<tr>
<td>checkComplexTables</td>
<td>Check Complex Tables</td>
<td>true/false</td>
<td>Specifies whether complex tables within the application are synchronized when the transmit configuration is used.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Note: Consider not selecting this attribute for transmit configurations that are intended for slower connection types.</td>
</tr>
</tbody>
</table>
Override Item | Setting | Acceptable Values | Description
--- | --- | --- | ---
keepAlivePeriod | Keep Alive Period | Value in seconds (the default is 0, meaning no keep-alive messages) | Determines how often a keep alive message is sent by either the client or the server. Keep alive messages are sent only for transmit configurations that are defined to "stay connected", which means that either background sending, and/or pushes must be enabled.  
- If pushes are enabled, the server sends the keep alive message to the client. For example, when keepAlivePeriod=45, the server sends a keep alive message to the client every 45 seconds.  
- Otherwise, if only background sending is enabled, the client sends keep alive messages to the server. For example, when keepAlivePeriod=60, the client sends a keep alive message to the server every 60 seconds.  
- If set to the default value of 0, no keep alive messages are sent.

Table 39: Session Attributes

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trackXmitEvents</td>
<td>Track Transmit Events</td>
<td>true/false</td>
<td>Whether to track transmit events.</td>
</tr>
<tr>
<td>stayLoggedIn</td>
<td>Stay Logged In</td>
<td></td>
<td>Keep the client user logged in and the application connected to the server to support real-time communications within the mobile application, which includes background sending and push behaviors. Set this attribute when an Agentry application requires a constant network connection to the back-end data.</td>
</tr>
<tr>
<td>promptOnLogin</td>
<td>Prompt on Log In</td>
<td>true/false</td>
<td>Display a prompt when the connection between the application and the server is lost, and the application attempts to reconnect.</td>
</tr>
<tr>
<td>promptOnLogout</td>
<td>Prompt on Log Out</td>
<td>true/false</td>
<td>Display a prompt when the Agentry application is logged out of the server.</td>
</tr>
<tr>
<td>serverInactiveTimeoutOverride</td>
<td>Inactive Timeout</td>
<td>Duration value in number of seconds.</td>
<td>Specifies the time limit, in seconds, to keep the application connected to the server, both when there is no data transmission activity between the client and server, and during an actual transmit. If you experience frequent disconnects, increase this number.</td>
</tr>
<tr>
<td>attemptOnlinePeriod</td>
<td>When Off-line</td>
<td>-1: Stay offline  1+: Attempt to work online every [1+] seconds</td>
<td>If the connection to the server is lost, indicates whether to stay offline or attempt to reconnect at the specified interval.</td>
</tr>
</tbody>
</table>
### Override Item

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>onlineRetryAttempts</td>
<td>Attempts</td>
<td>Number of attempts to connect to server.</td>
<td>If you select to reconnect when offline, indicate the number of attempts to reconnect. If the number of reconnect attempts fail, the way the application responds depends on the Transmit Configuration attributes you defined. For example, the application can display a prompt or switch to the failover transmit configuration.</td>
</tr>
<tr>
<td>reconnectAttemptPeriod</td>
<td>Reconnect Attempt Period</td>
<td>Duration value, in number of seconds</td>
<td>If online, indicates how often the client should try to reconnect.</td>
</tr>
<tr>
<td>retryPeriod</td>
<td>Retry Period</td>
<td>Duration value, in number of seconds</td>
<td>If the connection to the server is lost, indicates how long to attempt to reconnect.</td>
</tr>
<tr>
<td>synchronousPlaySuccessSound</td>
<td>Synchronous Play Success Sound</td>
<td>true/false</td>
<td>Whether to play a sound on synchronous success.</td>
</tr>
<tr>
<td>synchronousPlayFailureSound</td>
<td>Synchronous Play Failure Sound</td>
<td>true/false</td>
<td>Whether to play a sound on synchronous failure.</td>
</tr>
<tr>
<td>synchronousFailureSound</td>
<td>Synchronous Failure Sound</td>
<td>string</td>
<td>Identifies the file name for the failure sound to play, if enabled. If left blank, a default error beep is used.</td>
</tr>
<tr>
<td>synchronousSuccessSound</td>
<td>Synchronous Success Sound</td>
<td>string</td>
<td>Identifies the file name of the success sound to play, if enabled. If left blank, a default &quot;OK&quot; beep is used.</td>
</tr>
</tbody>
</table>

### Note

The synchronous transmit settings refer to communication that results from a transmit step in an Agentry action, not background sending nor pushes.

### Table 40: Background Sending

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>backgroundConnection</td>
<td>Allow background sending from client</td>
<td>true/false</td>
<td>Whether to allow Agentry clients to send transactions in the background. When background sending is enabled, clients remain logged into the server, and send completed transactions in the background while a user continues to work online. When a user works offline, transactions are sent when a connection is available.</td>
</tr>
<tr>
<td>retryPeriod</td>
<td>Retry Period</td>
<td>Duration in number of seconds</td>
<td>If background sending is enabled, specifies the amount of time, in seconds, to wait before attempting to resend a failed message.</td>
</tr>
<tr>
<td>Override Item</td>
<td>Setting</td>
<td>Acceptable Values</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-----------------------</td>
<td>-------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>retryAttempts</td>
<td>Retry Attempts</td>
<td>Number of back-</td>
<td>If background sending is enabled, specifies the number of times the client should attempt to resend a failed message.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ground sending</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>attempts to</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>connect to server</td>
<td></td>
</tr>
</tbody>
</table>

**Table 41: Push**

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Allow server to push</td>
<td>None</td>
<td>Allow the server to push data to the application. A definition for the Pushes module must exist for the application. Users connecting to the server are logged in as push users.</td>
</tr>
<tr>
<td></td>
<td>data to client</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note**

You must set this attribute before you can set the other push attributes for the transmit configuration.

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Acceptable Values</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>retryPeriod</td>
<td>Retry Period</td>
<td>None</td>
<td>If “Allow server to push data to client” is selected, this attribute specifies how long the server waits before attempting to resend data for a push when a failure occurs.</td>
</tr>
<tr>
<td>attempts</td>
<td>Attempts</td>
<td>None</td>
<td>If “Allow server to push data to client” is selected, specifies how many attempts to push data to the application when a failure occurs.</td>
</tr>
</tbody>
</table>

**Table 42: Modem Settings**

<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Modem Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networkCheck</td>
<td>Check for Modem</td>
<td>true/false</td>
<td>Whether the Agentry application checks for a modem connection when using the transmit configuration prior to beginning the transmit.</td>
</tr>
<tr>
<td></td>
<td>Connection</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note**

You must set this attribute before you can set the other modem connection attributes for the transmit configuration.
<table>
<thead>
<tr>
<th>Override Item</th>
<th>Setting</th>
<th>Modem Attribute</th>
<th>Description</th>
</tr>
</thead>
</table>
| networkConnection-Name        | Connection Name                            | Name of Windows network connection.   | Two options:  
* Any dial-up connection requires a user to establish the network connection manually outside the mobile application before beginning the transmit. The remaining modem connection attributes are disabled.  
* Name of any Windows network connection that is configured on the client device. This option uses the settings of the specified connection to establish the modem connection to the network.                                                                                                                                                                                                                     |
| connectNetwork                | If Not Connected                           | true/false                            | Specifies that the application attempt to create a connection using the specified Windows network connection when there is no current connection. If this option is not selected, the remaining modem connection attributes are disabled.                                                                                                                                                                                                                                                                                                                                 |
| networkConnection-Prompt      | Connect Prompt                             | Any printable text. Carriage returns not allowed. Text auto-wrap on the client. | Provides instructions to the user on the client prior to attempting to create a modem connection. Leave blank if no message is required.                                                                                                                                                                                                                                                                                                                               |
| dialUsername-Prompt           | Prompt User for Dial-up Username           | true/false                            | Prompts the user to enter a user name for the network connection, which is used as the login name for the network connection once the modem’s handshaking processes are successful. If this option is not selected, the Agentry application login is used.                                                                                                                                                                           |
| dialPasswordPrompt            | Prompt User for Dial-up Password           | true/false                            | Prompts the user to enter a password for the network connection. If this option is not selected, the Agentry application password is used.                                                                                                                                                                                                                                                                                                                                            |
| modemInitWait                 | Modem Init Wait                            | Duration value, in milliseconds       | Specifies the amount of time, in milliseconds, to wait for the client device’s modem to initialize before beginning the dial-up process.                                                                                                                                                                                                                                                                                                                                                                     |
| postConnectWait               | Post-connect Wait                          | Duration value, in milliseconds       | Specifies the amount of time, in milliseconds, to wait after the network connection is made before beginning the transmit process between the client and server.                                                                                                                                                                                                                                                                                                                                 |
| closeConnectionPeriod         | Close Connection                           | Duration value, in seconds.           | Specifies when to close the modem connection once no more data is being transmitted between client and server:  
* Never  
* If the connection is idle for the specified number of minutes and seconds.                                                                                                                                                                                                                                                                                                                                                               |
| remindToTurnOnModem           | Remind to Turn On Modem                    | true/false                            | Whether to remind user to turn on modem.                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| remindToTurnOffModem          | Remind to Turn Off Modem                   | true/false                            | Whether to remind user to turn off modem.                                                                                                                                                                                                                                                                                                                                                                                                                                  |
1.4.3.6 Managing and Monitoring Applications

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

In Mobile Service for Development and Operations cockpit, you can view the properties of Fiori applications and connections that were developed using SAP Cloud Platform mobile service for app and device management and imported into SAP Cloud Platform mobile service for development and operations, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Managing Applications [page 121]
Manage multiple native, hybrid, and Web applications from a single location.

Managing Registrations and Users [page 128]
Manage multiple application registrations from a single location. Registrations are associated with an authenticated or an anonymous user on one or more devices.

Exporting and Importing Application Configurations [page 131]
The ability to import and export application configurations enables you to copy a configuration from one environment to another. For example, you can export an application configuration from the testing environment, and import it to the production environment.

Managing Destinations [page 134]
Manage multiple destinations from a single location.

Manage SAP Content to Go Card Types [page 144]

Managing the Agentry Server Instance [page 153]
The Agentry tile is available for accounts that have the Agentry capability enabled. Use the Mobile Services cockpit to manage Agentry features that are linked to your account.

Sending a Push Notification [page 155]
Send a push notification to all users of a push-enabled application, or a subgroup of recipients (such as all iOS devices in a particular time zone).

Client and Server Data Report [page 158]
You can use Mobile Service for Development and Operations cockpit to view aggregated data for native, hybrid, and Web applications and its clients. Data reports appear in graphical form and provide summaries of client and server sessions.

Application Logs and Trace Files [page 161]
Set the verbosity for application and component logging, and define how long to keep logs and trace files. You can view all application logs or a subset of your choice, and drill down to view detailed log and trace information if available.

Tracing Network Activity [page 169]
Application users can trace network activity based on user name, connection, application, or content type. You can download tracing reports to either a .zip or a .har file.

Viewing the User Audit Log [page 171]
Use audit log information to view user actions over the most recent six months.

Manage Signing Profiles [page 172]
You can create signing profiles for Android and iOS applications. As an administrator, you can control access to the signing profiles that are available for an account.

Monitoring Hybrid Application Versions [page 174]
When multiple versions of a hybrid application are running on registered devices, you can see the percentage of devices that are running each version of the application.

SAP License Auditing [page 175]

The SAP Cloud Platform mobile service for development and operations License Audit feature enables you to generate an SAP audit measurement file in accordance with the SAP License Auditing process.

Related Information

Configuring SAP Content to Go Applications [page 95]
Creating Card Types [page 145]
Editing a Card Type [page 149]
deleting a Card Type [page 149]
Defining Categories [page 150]
Defining Actions [page 151]

1.4.3.6.1 Managing Applications

Manage multiple native, hybrid, and Web applications from a single location.

Context

You can add, edit, or delete applications; ping a selected application for a back-end connection; export an application; and publish or unpublish an application. You can view all applications, or retrieve a filtered subset of applications. You can view the applications using a list view, or a tile view. From the list of applications, you can drill down to see summary, registration, and usage information for a particular application.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid.

   You can view a list of applications with summary information that includes application ID, name, type, vendor, security configuration, creation date, and actions. You can switch the presentation between the tile view and list view. Use the filter and sort options to locate the applications in which you are interested.

2. Identify an application of interest, and, under Actions, click the Actions icon.
Applications are listed according to their origin policies. An origin policy specifies the access rights for applications and connections that are created in another origin (which typically means an origin other than SAP Cloud Platform mobile service for development and operations). Some read-only actions or fields may be dimmed, or not available.

You may see an additional action, Push if a push-enabled application is selected, and the administrator or user is assigned the Notification User role.

3. Select Overview to view the information, registration, and usage details for the selected application.
   - The Information tab is a summary of settings for the selected application. You can easily copy URLs to paste elsewhere, avoiding typing errors and ensuring accuracy. See Defining Applications to set the information on this tab.
   - Select Registrations to view details about valid application registrations. You can filter registrations by search field, sort information by column, select the columns that appear, and refresh the data.
   - Select Usage to see usage statistics for the application. Choose your options:
     - Select the Time Frame filter for the registration time period, including Today, Yesterday, Last 7 Days, Last 4 Weeks, Last 3 Months, and Last 12 Months.
     - Indicate whether to show data in bar-chart format (column or line), or table-chart format.

4. Use Publish and Unpublish to manage the application.
5. Use the back arrow to navigate back to the list or tile view; or select a menu option—such as Configure, Push, or Export—to continue with the selected application.

Editing an Application [page 123]
Edit an existing application from the application list.

Deleting an Application [page 124]
Delete the application from the application list.
Pinging a Back-End Connection [page 124]
Test that back-end connections (one primary and multiple secondary endpoints) are reachable for the selected application.

Managing Application Versions [page 125]
Administrators can activate or deactivate specific versions of an application. If an application version is not active, users must upgrade it to an active version. Multiple versions of an application can be active at the same time.

Exporting Data [page 125]
SAP Cloud Platform mobile service for development and operations allows you to export a tenant database from the cloud to a local CSV file.

Modifying Cloud Build Settings [page 126]
Modify an existing Fiori or Hybrid app that already has a binary built.

1.4.3.6.1.1 Editing an Application

Edit an existing application from the application list.

Context
In Mobile Service for Development and Operations cockpit, you can view the properties of Fiori applications and connections that were developed using SAP Cloud Platform mobile service for app and device management and imported into SAP Cloud Platform mobile service for development and operations, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications ➤ Native/ Hybrid ➤

2. Identify an application, and, under Actions, click ➤ Configure ➤

3. Select a tab, and make changes.
4. Save your changes.
1.4.3.6.1.2 Deleting an Application

Delete the application from the application list.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** > **Native/Hybrid**.
2. Identify an application, and, under **Actions**, click **Delete**.
3. Click **OK** to confirm.

**Note**

Once the application has been deleted, users cannot use it. All existing logs and traces are deleted and cannot be retrieved.

1.4.3.6.1.3 Pinging a Back-End Connection

Test that back-end connections (one primary and multiple secondary endpoints) are reachable for the selected application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** > **Native/Hybrid**.
2. Identify an application, and, under **Actions**, click **Ping** to view the back-end connection status.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connection Name</td>
<td>Back-end connection name</td>
</tr>
<tr>
<td>Backend URL</td>
<td>Endpoint URL</td>
</tr>
<tr>
<td>Ping Result</td>
<td>Current state of the back-end connection</td>
</tr>
</tbody>
</table>
Table 44: Status Descriptions

<table>
<thead>
<tr>
<th>Message</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ping Successful</td>
<td>The back end is reachable.</td>
</tr>
<tr>
<td>&lt;Error code: Error message&gt;</td>
<td>The back end is not reachable.</td>
</tr>
</tbody>
</table>

3. Click OK.

1.4.3.6.1.4 Managing Application Versions

Administrators can activate or deactivate specific versions of an application. If an application version is not active, users must upgrade it to an active version. Multiple versions of an application can be active at the same time.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications.
2. Identify an application, and select Configure.
4. To add a version, under Application Version Settings, select the Add new version icon, and in the new dialog, enter:
   - Version – application version number, for example, 1.0.
   - Active – select Yes or No.
   - Description – optional description of the application version.
5. (Optional) To delete an application version, select it, and click the Remove selected version icon.

1.4.3.6.1.5 Exporting Data

SAP Cloud Platform mobile service for development and operations allows you to export a tenant database from the cloud to a local CSV file.

Exporting data is a role-based feature, and you should have one of the following roles to perform this task:
- Administrator
- Helpdesk

Exporting a tenant database from the cloud to a local CSV file can be performed in the following systems:
- SAP Cloud Platform mobile service for development and operations
- Microservices / Storage
Microservices / Octane

You can export a tenant database from the cloud to a local CSV file by using the following command: (??)

/mobileservices/origin/{origin}/exportdata/v1/admin

Example

You can use the following link to export the data:

https://<HMC base URL>/mobileservices/origin/hcpms/exportdata/v1/admin

Response Status and Error Codes

Table 45:

<table>
<thead>
<tr>
<th>Code</th>
<th>Response Body</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>&lt;exported data&gt;</td>
<td>Prompts the user to download the .CSV file.</td>
</tr>
<tr>
<td>403</td>
<td></td>
<td>The user does not have the Administrator or Helpdesk role.</td>
</tr>
</tbody>
</table>

1.4.3.6.1.6 Modifying Cloud Build Settings

Modify an existing Fiori or Hybrid app that already has a binary built.

Context

View the details for each binary build, and view and download build logs. You can also modify build settings and rebuild a binary, and you can download a binary.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications Native/Hybrid
2. Select an application, and then select Cloud Build under Assigned Features (or add it first).
3. Select the Build History tab. You can view the build history for each build package created for the application.
Table 46: Cloud Build History

<table>
<thead>
<tr>
<th>Item</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>Current state of the build, such as Building, Build Failed, and Build Complete.</td>
</tr>
<tr>
<td>Debug Enabled</td>
<td>Whether the debugging tool is enabled.</td>
</tr>
<tr>
<td>Operating System</td>
<td>The operating system of the build, such as iOS or Android.</td>
</tr>
<tr>
<td>Form Factor</td>
<td>The device type, such as tablet or phone.</td>
</tr>
<tr>
<td>App Version</td>
<td>The application version.</td>
</tr>
<tr>
<td>Built On</td>
<td>The date of the build, in the format YYYY-MM-DD T HH:MM:SS SubS.</td>
</tr>
<tr>
<td>Certificate Expiration</td>
<td>The date the certificate expires.</td>
</tr>
<tr>
<td>Actions</td>
<td>Actions you can take, such as Delete to delete the download binary.</td>
</tr>
</tbody>
</table>

4. (Optional) Select a row to view details of a build package.
   a. View the build settings:
      ○ App Version / Operating System – the application version and operating system type, such as 1.2/iOS.
      ○ Built On/Certificate Expiration – the build date and the certificate expiration date, such as YYYY-mm-DD T HH:MM:SS SubS.
      ○ Minimum OS Version – the minimum operating system version, such as 10.0.
   b. View the build options selected for the build:
      ○ Disable Passcode Screen – whether the passcode screen was disabled.
      ○ Disable Privacy Screen – whether the privacy screen was disabled.
      ○ Verbose Logging – whether verbose logging was enabled.
      ○ Debug-enabled Binaries – whether binaries were built with debug enabled.
      ○ Clear cached content prior to initializing build – whether the cache was cleaned up before the build started.
   c. Under Build Logs, you can view the build log details on the screen, or download the log to a local directory.
   d. Click Download Binary to download the binary to a local directory. Click OK to confirm. When ready, you can post the binary to a download site.

5. (Optional) To create a new build package for the application, click Build from the Build History tab.
   a. On the Build screen, select the platforms to build, such as Android and iOS.
   b. For each platform:
      ○ Signing Profile – select a signing profile.
      ○ Create Debug-enabled Binaries – whether binaries were created with debug capabilities
      ○ Minimum OS Version – select the minimum operating system version required.
      ○ Build Options – select the build options to use for the build (described in Step 4).
c. Click OK to initiate the build. A new build history row is added.

1.4.3.6.2 Managing Registrations and Users

Manage multiple application registrations from a single location. Registrations are associated with an authenticated or an anonymous user on one or more devices.

Context

View information for all application registrations, or retrieve a filtered subset of registrations. From the list of registrations, you can see application and user details, HTTP traffic between applications and back ends, and log information for offline communication. For hybrid applications, if the developer has implemented logger code in the application, you can upload and view client logs. You can also delete registrations.

When a user logs in from a device, or is logged in anonymously by a client application, he or she is authenticated using the security configuration of the application. If authentication is successful, SAP Cloud Platform mobile service for development and operations generates a registration identifier for the application +user+device combination, and creates a record in the persistent storage database, which enables the device application to consume data and services.

Note

Customers subscribe to mobile services based on both the user count and the bandwidth between devices and SAP Cloud Platform mobile service for development and operations. SAP measures the user count and bandwidth annually, and notifies the technical contact if licensed resources have been exceeded.

SAP Cloud Platform mobile service for development and operations sends the registration identifier to the device application. For all subsequent requests, such as accessing data in the back-end data source, the device client sends the registration ID to the mobile service.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics > User Registrations.

Note

You can change the columns that initially appear in a later step.
Table 47: Column Descriptions

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registration ID</td>
<td>Unique identifier provided by the client application, or system-generated application registration ID.</td>
</tr>
<tr>
<td>User Name</td>
<td>User of the registered application.</td>
</tr>
<tr>
<td>Application ID</td>
<td>Unique identifier for the application. SAP recommends that IDs use reverse-domain notation; for example, com.sap.*.</td>
</tr>
<tr>
<td></td>
<td>This is the application or bundle identifier that the application developer assigns or generates during application development. The administrator uses the application ID to register the application to SAP Cloud Platform mobile service for development and operations, and the client application code uses the application ID when sending requests to the server.</td>
</tr>
<tr>
<td>Device Type</td>
<td>Type of device, such as Android or iPhone, sent by the device during registration/onboarding. “Unknown” indicates that the device type cannot be detected.</td>
</tr>
<tr>
<td>Beta Tester</td>
<td>Yes means the user is a Beta tester; No means the user is not. Users who are defined as Beta testers can test Kapsel apps that are in the staged state. See Uploading and Deploying Hybrid Apps [page 49].</td>
</tr>
<tr>
<td>Last Connection</td>
<td>The date, time, and time zone when the application was registered, in the format YYYY-MM-DD HH:MM.</td>
</tr>
<tr>
<td>Usage Upload</td>
<td>The date, time, and time zone when the usage upload consent was given or revoked, in the format YYYY-MM-DD HH:MM.</td>
</tr>
<tr>
<td>Log Settings</td>
<td>The configured log settings for the client.</td>
</tr>
</tbody>
</table>

2. (Optional) Select the following filtering options to view a subset of application registrations:
   - Application ID
   - Device Type
   - Registration Time Frame
   - User name

3. Click Go.

4. To customize the column display, click and select the columns to appear.

5. To customize sorting, click and select the sort field. You can also select ascending or descending order. Sort results are based on the columns that appear, including customized table columns.

Deleting Registrations and Users [page 130]
Delete one or more registrations. Each registration entry includes a registration ID, an application ID, a device type, and a user name.

Managing Client Database Uploads [page 130]
Manage client database files that are uploaded from device users. You can download the file for analysis, or drill down to an individual application to troubleshoot and review uploaded files.
1.4.3.6.2.1 Deleting Registrations and Users

Delete one or more registrations. Each registration entry includes a registration ID, an application ID, a device type, and a user name.

Context

Deleting a registration is useful when a registration is orphaned; this can occur if a user is no longer using a device application, or is using a new mobile device that requires its own registration. You may also want to delete a registration if you are troubleshooting configuration issues. After a registration is deleted, the user must reregister the application to use it on the device.

Procedure

1. From Mobile Service for Development and Operations cockpit, select Analytics User Registrations.
2. Click the icon next to the username that you want to delete.

1.4.3.6.2.2 Managing Client Database Uploads

Manage client database files that are uploaded from device users. You can download the file for analysis, or drill down to an individual application to troubleshoot and review uploaded files.

Context

Client database files can be uploaded only by application users for applications for which an upload policy is enabled.
Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid Applications.
2. Select an application.
3. In the Info tab, click icon in the Assigned Features section.
4. In the Add feature window, select Offline and click OK.
5. Select Client Databases tab.
6. (Optional) Select filtering options to view a subset of application with uploaded client databases, including unique ID, application ID, device ID, or creator name.
7. At any time, click Refresh, to update the current list of applications with uploaded client databases.
8. To sort results based on application ID, device ID, creator, file size (bytes), and creation time, click and select the sort by field. You can also select ascending or descending as the sort order.
9. To download the client database file for analysis, click Download and confirm. Identify a program with which to open the file, and provide a temporary name.
10. When finished, select one or more applications with uploaded client databases, click Delete and confirm. If you do not manually delete a database file, it is deleted automatically according to the client database policy.

1.4.3.6.3 Exporting and Importing Application Configurations

The ability to import and export application configurations enables you to copy a configuration from one environment to another. For example, you can export an application configuration from the testing environment, and import it to the production environment.

Exporting Application Configurations [page 132]
Export an application configuration zip file to your local system, retaining many of its settings. You can use the export feature to create a back-up of the application configuration, and as a prerequisite for importing the application configuration to SAP Cloud Platform mobile service for development and operations.

Importing Application Configurations [page 133]
Import an application configuration from one SAP Cloud Platform mobile service for development and operations environment to another. Many configuration settings are retained, but you must reconfigure some application settings for the target server environment.
1.4.3.6.3.1 Exporting Application Configurations

Export an application configuration zip file to your local system, retaining many of its settings. You can use the export feature to create a backup of the application configuration, and as a prerequisite for importing the application configuration to SAP Cloud Platform mobile service for development and operations.

Prerequisites

Before exporting an application to your local system, verify that the application status is consistent (marked in green).

Context

Only the current version of an application is exported; new versions are not exported.

Procedure

1. In Mobile Service for Development and Operations cockpit, click Applications.
2. Select the application, and select Export.
3. Select the configuration to export, and click Export.

The application configuration file is downloaded to the <appname>_<version>.zip file in the default location, which is specified in the browser.

- <appname>_<version>.zip contains a Common subfolder with the <appname>_<version>.smpconfig file that contains all the application information.
- Passwords are not exported.

Note

You can export the application configuration to a shared directory to make it available to all the mobile service for development and operations systems that have access to the shared directory.
1.4.3.6.3.2 Importing Application Configurations

Import an application configuration from one SAP Cloud Platform mobile service for development and operations environment to another. Many configuration settings are retained, but you must reconfigure some application settings for the target server environment.

Prerequisites

Verify that the application *.zip file you want to import is available on the SAP Cloud Platform mobile service for development and operations network.

Context

- Importing an application configuration does not necessarily make it ready for use. Administrators must review the imported application, and if necessary, make adjustments to its configuration.
- For hybrid apps that use the AppUpdate plugin, manage the application version using Applications → App Spec Setting.
- If you make changes to an exported application configuration *.zip file, you cannot import it.

Note

You can import a Fiori application that was developed with SAP Cloud Platform mobile service for app and device management, but you cannot edit its properties in Mobile Service for Development and Operations cockpit.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications.
2. Click the Import icon.
3. Click Browse, and select the application configuration *.zip file.
   - To import an application with the same ID as an existing application, select Overwrite existing application; otherwise, you see the error message Application already exists.
   - If you import an application with a secondary connection that already exists in the target system, the connection is not overwritten.
   - Passwords such as those for APNS, BES, and anonymous user settings are not imported, so the application is marked inconsistent.
   - If any resource bundles were exported, they are uploaded during import.
4. Click Import.

When the import completes, you see the message Application Created.
Next Steps

Configure the application for the target server environment.

1.4.3.6.4 Managing Destinations

Manage multiple destinations from a single location.

Context

You can maintain destination types:

- Mobile Destination – destination to back-end or Web service.
- Cloud Platform Destination – use an existing SAP Cloud Platform destination, without having to enter it manually. Some of the connection level configuration settings (keystore, SSO, etc.) are not available for all destinations. You must create and configure destinations in the SAP Cloud Platform.

You can define, edit, and delete destinations, except where restricted; test a destination using Ping, search for a destination, and sort destinations based on Name and URL. You can view a summary of settings for a selected destination.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Destinations.
2. View current mobile, Fiori, and cloud platform destinations.

Table 48: Destinations

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The destination name.</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Platform Destination Name</td>
<td>For Cloud Platform Destination types, this is the original cloud platform destination name (it may differ from the destination name used in Mobile Service for Development and Operations cockpit). For Mobile Destination types, this is not used (Not Applicable). For Fiori Destination types, this is not used (Not Applicable).</td>
</tr>
<tr>
<td>URL</td>
<td>The destination URL.</td>
</tr>
<tr>
<td>Proxy Type</td>
<td>The proxy type used for the destination - Internet, or On Premise (Cloud Connector).</td>
</tr>
<tr>
<td>SSO Mechanism/Authentication</td>
<td>The single sign-on or authentication security methods employed for the destination.</td>
</tr>
<tr>
<td>Actions</td>
<td>The actions available, such as test an OData application destination, ping a back-end connection, or delete a connection</td>
</tr>
</tbody>
</table>

3. (Optional) To display a subset of destinations, use **Search**.
4. (Optional) To view details about a specific destination or edit the destination, select the destination.

**Creating a Destination** [page 135]

Define a new destination to a data source or service.

**Editing a Destination** [page 143]

Modify settings for an existing destination.

**Pinging a Destination** [page 143]

Test whether a destination is accessible.

**Deleting a Destination** [page 144]

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

### 1.4.3.6.4.1 Creating a Destination

Define a new destination to a data source or service.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Destinations**, and click **New**.
2. In **Create Destination**, select the destination type:
- Mobile Destination – destination to back-end or Web service.
- Cloud Platform Destination – use an existing SAP Cloud Platform destination, without having to enter it manually. Some of the connection level configuration settings (keystore, SSO, etc.) are not available for all destinations. You must create and configure destinations in the SAP Cloud Platform.

**Note**
The default Fiori Client destination cannot be edited or deleted.

3. Give the destination a name.
   - If you selected Mobile Destination or Fiori Destination in Step 2:
     1. In **Destination Name**, enter a new destination name.

     **Note**
     Fiori destination name should start with an alphabet, can contain only alphanumeric characters, underscores and can be as many as 32 characters long

     2. Click **Next**.
   - If you selected Cloud Platform Destination in Step 2:
     1. In **Platform Destination Name**, select an existing cloud platform destination.
     2. In **Destination Name**, enter a unique destination name to associate with the cloud platform destination name.
     3. Click **Next**.

4. Use the Create Destination wizard to configure destination values. The wizard is dynamic, so the options that appear and their order may differ, depending on your selections. Use the following information to help make your entries, and click **Next** to progress through the wizard.

**For Mobile and Cloud Platform Destinations:**

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>URL that the application uses to access business data on the back-end system or service. If the URL points to a service, it must include the document destination that you assign to the service. The SAP Cloud Platform integration service OData provisioning capability supports REST, OData, and SOAP data sources. It also provides Web tools you can use to convert data sources to OData services, and deploy those services. You can use OData service URLs when creating back-end connections in the cockpit for the development and operations mobile service.</td>
</tr>
</tbody>
</table>

**Note**
The current implementation supports only basic authentication; it does not support application-to-application single sign-on (SSO). Internet is the default proxy type for OData provisioning that SAP Cloud Platform integration service provides.
In the cockpit for the development and operations mobile service, use the catalog viewer to select the provisioned OData Service URL:

1. Select Add.
2. Select a destination.
3. Select the service URL.

**Note**

The OData provisioning capability in both SAP Cloud Platform integration service and Back-End Connection Catalog Viewer is in Beta mode.

The URL must include a trailing forward slash. This is especially important when configuring the application with security such as SSO tokens and certificates, and when Rewrite URL is enabled.

Examples:

- `http://mobile<account_name>-<tenantname>.hana.ondemand.com/sap/opu/odata/RMTSAMPLE/`

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
</tbody>
</table>

### Proxy Type

Select either:

- **Internet – HTTP destinations** use an Internet proxy. The back-end URL must be a publicly accessible Internet URL. Back-end systems are in the public domain and accessible to everyone, for example, `http://www.google.com`.
- **On Premise (Cloud Connector)** – back-end systems are behind a firewall, and HTTP destinations use SAP cloud connector to connect to these systems. The back-end URL must be an SAP cloud connector URL (that is, the URL must be a virtual host).

### Maximum Connections

(Optional) The maximum number of connections that this application can use for connection pooling. Valid values are 0–9999.

Factors to consider are:

- Expected number of concurrent application users
- Acceptable load for the back-end system

To disable connection pooling, set the value to 0. This creates a new connection for each new request, which may increase processing times. SAP recommends that you disable connection pooling only if the back-end system does not support pooled connections.

### Timeout (in milliseconds)

The number of milliseconds before the connection times out. If set to 0, the JVM timeout value is used.

### Online Request Threshold

The threshold value to throttle incoming online requests per second for a connection. Leave blank (default) or set to 0 to remove threshold. Set a threshold value from 1 – 2147483647.
### Field | Value
--- | ---
Rewrite Mode | **Note**
 | To enable applications that use external back ends to run offline, you must select either *Rewrite URL* or *Rewrite URL on Back End*. Select one of:
 | ○ *Rewrite URL* – in request and response messages, the mobile service for development and operations replaces all back-end URLs with the mobile service URL. The Rewrite URL format for Web-type applications is `https://<mobileServiceHost>/<back-end_connection_ID>?X-SMP-APPID=<applicationID>`.  
 | ○ *Rewrite URL on Back End* – the back end rewrites the URLs. The mobile service for development and operations forwards the host name and port to the back end in an HTTP header, and the back end creates the URL to retrieve back-end resources. To expose the full URL to clients, the mobile service passes the endpoint in the `X-SMP-ENDPOINTNAME` header. The URL format for Web applications is `https://<host>/<back-end_path>?X-SMP-APPID=<applicationID>`.  
 | ○ *No Rewriting* – request and response messages are not modified. The mobile service for development and operations passes messages directly between clients and the back end. The URL format for Web applications is `https://<mobileServiceHost>/<back-end_connection_ID>?X-SMP-APPID=<applicationID>`.  
 | **Note**
 | ○ The mobile service for development and operations does not provide the functionality to use No Rewriting mode to support external back ends for offline usage.  
 | ○ SAP Content to Go applications should use only *No Rewriting* mode.  
 | ○ *Custom Rewrite URL* – for request and response messages, you can define a search string and a replacement string, which need not be URLs. For more details about the different rewrite mode options, see the *Rewrite Modes* topic.

Relative Rewrite Paths | Enter a comma-delimited list of relative URLs, for example, `/sap/bc`, `/sap/public/bc`. If an application requires data from a back end that uses relative URLs, define them here. The mobile service for development and operations rewrites the relative URLs to include the connection name, enabling access to the back-end data.  
 | For example, a Web service application requests an HTML page named `abc.html`, which contains the relative URLs `/sap/bc` and `/sap/public/bc` in its `src` or `href` tags. When a request is made, the relative URLs contained in the response are rewritten, so that subsequent requests (to these relative URLs) can be processed correctly. For example, if "webApp" is the connection name, and the response contains the relative URLs `/sap/bc`, `/sap/public/bc`, these are changed to `/webApp/sap/bc`, `/webApp/sap/public/bc`. |
### Field Values

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCIM Path</td>
<td>(Optional) Modify the SCIM path prefix (protocol, host, or port). The default value is “/”. If you update the path, you must also select Rewrite Override Global Mobile Service SCIM Configuration; otherwise, the path is ignored.</td>
</tr>
<tr>
<td>Propagate User Name</td>
<td>(Not applicable when application Security Configuration is set to None) When enabled, the back end uses information in the X-SMP-ENDUSERNAME &lt;user name&gt; header to identify the user who sent the request. See HTTP Headers Used to Propagate User IDs. By default, this option is disabled.</td>
</tr>
<tr>
<td>Virus Scans Inbound Traffic</td>
<td>The server performs a virus check scan for the incoming data. Rewrite URL</td>
</tr>
<tr>
<td>Virus Scans Outbound Traffic</td>
<td>The server performs a virus check scan for the outgoing data.</td>
</tr>
</tbody>
</table>

**For Fiori Destinations:**

Table 49: Fiori Destination Settings

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>URL</td>
<td>The Fiori destination URL. Fiori Client Destination supports only On Premise proxy type, hence only HTTP protocol can be used.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>To configure Fiori Client using its destination URL, and publish to the Discovery Service, specify the URL value as fiori_serverhost:port.</td>
</tr>
<tr>
<td>SAP Client ID</td>
<td>Enter the SAP Cloud Id that is embedded in the URL, such as 001.</td>
</tr>
<tr>
<td>SSO Mechanism</td>
<td>To configure Fiori Client using its destination URL, and publish to the Discovery Service, select Basic Authentication or Principal Propagation.</td>
</tr>
<tr>
<td></td>
<td><strong>Note</strong></td>
</tr>
<tr>
<td></td>
<td>For Basic Authentication SCIM Path needs to be specified to access the back-end system.</td>
</tr>
</tbody>
</table>

5. (Optional) In **SSO Mechanism**, select a single sign-on option from the list of available options.

### SSO Mechanism

<table>
<thead>
<tr>
<th>SSO Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application-to-Application SSO</td>
<td>Enables mobile services to propagate user identities to other applications, which are consumed (deployed or subscribed) in the same SAP Cloud Platform account. A user identity is propagated to the application that is specified in the URL.</td>
</tr>
</tbody>
</table>
**SSO Mechanism** | **Description**
---|---
Tip | To find the URL, open SAP Cloud Platform Cockpit, and on the **Overview** page, see **Application URL**.

Requirements:
- The proxy type for the destination must be Internet.
- The application that is receiving the SSO connection (receiving application) must be consumable from the same SAP Cloud Platform account – either through deployment or a subscription.
- The receiving application must be either a Java or an HTML5 application.
- When developing a Java application, see **Enabling Authentication** for information about enabling the application to accept application-to-application SSO. Configure your account to allow principal propagation; see the "Specifying Custom Local Provider Settings" section in **ID Federation with the Corporate Identity Provider**.

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

- To configure ApptoAppSSO for an application not hosted on the same SAP Cloud Platform account; see the `saml2_audience` section in **Application-to-Application SSO Authentication**.

**SAPAssertionSSO** | Configure the back-end system to accept SAP assertion tickets that are signed by a trusted X.509 DSA key pair, and define these properties.

**Note**
You can either select a PKCS #12 file, in which case, the certificate and signing key properties are populated automatically, or you can define these properties manually. The certificate must be a Digital Signature Algorithm (DSA).

- **PKCS #12 File** – click **Browse** and select a PKCS #12 file that contains the certificate and signing key.
- **File Password** – password to access the PKCS #12 file.
- **Issuer SID** – system ID of the certificate issuer; must be trusted by the back-end system.
- **Issuer Client** – client ID of the certificate issuer; must be trusted by the back-end system.
  - For a HANA XS back end, the same value as the Issuer SID.
- **Recipient SID** – back-end system ID.
  - For a HANA XS back end, you can find the SID at the top of the Admin console. In an SAP Gateway system, you can find the value in the connection properties, for example, SAPgui.
- **Recipient Client** – client ID of the back-end system.
  - For a HANA XS back end, enter **000**.
SSO Mechanism | Description
--- | ---
Certificate and Signing Key | generate the values using `openssl`:
1. Run:
   ```
   openssl dsaparam -out dsaparam.pem 1024
   openssl gendsa -out dsaprivkey.pem dsaparam.pem
   openssl req -new -x509 -key dsaprivkey.pem -out dsacert.pem
   openssl pkcs8 -topk8 -inform PEM -outform PEM -in dsaprivkey.pem -out dsaprivkeyonly.pem -nocrypt
   ```
2. Open the `dsacert.pem` file, and remove the `-----BEGIN CERTIFICATE-----` and `-----END CERTIFICATE-----` lines and the line-feed characters.
3. Copy and paste the contents of `dsacert.pem` into the `Certificate` field.
4. Open the `dsaprivkeyonly.pem` file, and remove the `-----BEGIN PRIVATE KEY-----` and `-----END PRIVATE KEY-----` lines and the line-feed characters.
5. Copy and paste the contents of `dsaprivkeyonly.pem` into the `Signing Key` field.

Principal Propagation | (If proxy type is OnPremise) Allows destinations to forward the identity of an on-demand user to SAP cloud connector, which then forwards it to the back-end system of the relevant on-premise system. An on-demand user need not provide his or her identity for each connection to an on-premise system when using SAP Cloud Connector.
You can use the principal propagation authentication type in hybrid cloud configurations that use SAP cloud connector, which can dynamically transform a user’s identity to an X.509 user certificate. SAP Gateway Java and SAP Gateway ABAP servers typically host the services that support principal propagation via SAP Cloud Connector. Configuration is required in SAP Cloud Connector as well as the SAP Gateway server to establish trust between the two for principal propagation to happen.
To learn more about configuring this trust relationship, see [SAP Cloud Platform Cloud Connector](https://help.hana.ondemand.com/help/frameset.htm?e6c7616abb5710148cfcf3e75d96596.html)

OAuth2SAMLBearerAssertion | Enables applications to use SAML assertions to access OAuth-protected resources.
Enter:
- **Audience** – intended assertion audience, which is verified by the target OAuth authorization server.
- **Client Key** – key that identifies the consumer to the authorization server.
- **Token Service URL** – URL of the OAuth server.
- **Token Service User** – user for basic authentication to the OAuth server.
- **Token Service Password** – password for the token service user.
- **System User** – user who requests an access token from the OAuth authorization server. If this property is not specified, the currently logged-in user is used.
- **Security Domain Qualifier** – security domain of the user for whom the access token is requested.
### SSO Mechanism

<table>
<thead>
<tr>
<th>Description</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>○ <strong>Company Identifier</strong> – company identifier.</td>
<td></td>
</tr>
<tr>
<td>○ <strong>SAML Assertion Issuer</strong> – issuer of the SAML assertion.</td>
<td></td>
</tr>
<tr>
<td>○ <strong>Authentication Class</strong> – value of the AuthnContextClassRef tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication. See the SAML 2.0 specification.</td>
<td></td>
</tr>
<tr>
<td>○ <strong>Name ID Format</strong> – value of the NameIdFormat tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication.</td>
<td></td>
</tr>
<tr>
<td>○ <strong>Define User as Name ID</strong> – if the value of the NameId tag is undefined, the generated SAML2 assertion uses the currently logged-in user.</td>
<td></td>
</tr>
</tbody>
</table>

### Basic Authentication

Enter a user name and password to access the back-end system.

If you do not provide a user name and password, and the mobile service for development and operations authenticates the end-user credentials using Basic, the user name and password credentials are propagated to the back end.

### Client Certification Authentication

Requires a client certificate that is signed by a certificate authority (CA) trusted by back-end systems. It uses a technical user certificate to perform mutual SSL authentication. The requirements include:

- Proxy type must be Internet.
- Back-end URL must use HTTPS.
- You must provide values for both the keystore and truststore parameters:
  - Keystore – contains the client certificate that should be used for mutual SSL authentication.
  - Truststore – contains the certificate authority that issued the server certificate for the back-end system, as a trusted certificate authority entry.

**Note**

The configured technical user certificate is always used to establish a mutual SSL connection. Even if the user is authenticated to the SAP Cloud Platform mobile service for development and operations using the certificate, it will not be forwarded to the back end.

### No Authentication

Back ends require no credentials for authentication. Your destination is granted direct access to the relevant on-premise service.

6. Click **Finish** to complete the configuration. A summary of configuration settings appears. You can click **Edit** to make any corrections.

### Related Information

- Accessing Services Through Proxy URLs [page 316]
- Rewrite Modes [page 60]
1.4.3.6.4.2 Editing a Destination

Modify settings for an existing destination.

Context

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

In Mobile Service for Development and Operations cockpit, you can view the properties of Fiori applications and connections that were developed using SAP Cloud Platform mobile service for app and device management and imported into SAP Cloud Platform mobile service for development and operations, but you cannot edit their properties; input fields and buttons are disabled or hidden.

Procedure

1. From the Mobile Service for Development and Operations cockpit, select Destinations, and select the destination to edit.
2. Click Edit.
3. In the Edit Destination window, edit the details as required.
   **Note**
   If the application is configured with an origin policy, some fields may not be available.
4. Click Finish.

1.4.3.6.4.3 Pinging a Destination

Test whether a destination is accessible.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Destinations.
2. Select the connection, and click Ping.
   You see either:
Ping Successful – connection is accessible.
Ping error – connection is not accessible. Click Show Details to see more information.

1.4.3.6.4.4 Deleting a Destination

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

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select Destinations, and identify the destination to delete.
2. Click Delete, and OK to confirm. You are prompted if the destination is in use and cannot be deleted.

1.4.3.6.5 Manage SAP Content to Go Card Types

You can manage SAP card types from the Mobile Service for Development and Operations cockpit. SAP Content to Go allows you to create, edit, or delete SAP card types. Use the Template Manager to create cards or customize cards using the existing templates.

- **Creating Card Types [page 145]**
  Create SAP card types from the Mobile Service for Development and Operations cockpit.

- **Defining Parameters [page 148]**
  Define parameters to be used in action URLs.

- **Editing a Card Type [page 149]**
  Edit SAP Card Types from the Mobile Service for Development and Operations cockpit.

- **Deleting a Card Type [page 149]**
  Delete SAP Card Types from the Mobile Service for Development and Operations cockpit.

- **Defining Categories [page 150]**
  Create a category and define actions for card types. Users can execute the defined actions.

- **Defining Actions [page 151]**
  Define an operation to be performed on the back end when the user executes an action.

- **Using iOS Spotlight Search [page 152]**
  Use iOS Spotlight Search to find SAP Content to Go cards without launching the app.

**Related Information**

Configuring SAP Content to Go Applications [page 95]
1.4.3.6.5.1 Creating Card Types

Create SAP card types from the Mobile Service for Development and Operations cockpit.

Procedure

1. In the Mobile Service for Development and Operations cockpit, select Mobile Applications SAP Content to Go.
2. Select Create a New SAP Card Type to define card templates or to create custom card templates.
3. Enter the card’s properties.

Table 50: SAP Card Properties

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Unique SAP card type name.</td>
</tr>
</tbody>
</table>
| Data Refresh Mode   | These values determine how often the server checks for changes on the card, and if changes are available, it supplies a new copy of the card when the client requests for one. This helps in improving optimization and performance.  
  ○ Keep up-to-date – the server checks for changes when the client requests updated content. If the server detects any changes, a new copy of the card is sent to the client.  
  ○ Use daily interval – the server checks for changes only if the card was last sent to the client before the time specified, and it’s currently later than this time. If the server detects changes, a new copy of the card is sent to the client. For example, if the data refresh time is 11am, and a client refreshes cards at 8:00am, the server does not check for changes; however, if the client refreshes at 11:30am, the server does send a new copy of the card if any changes are detected.  
  ○ Use custom interval – the server checks for changes only if the card is older than the data refresh time. If the server detects changes, a new copy of the card is sent to the client. |
| Refresh Interval    | Specify the weeks, days, hours, and minutes at which custom updates occur. |
| Data Refresh Time   | Specify the time at which daily updates occur.                        |
| Back End            | Choose from an existing configured back end. The application accesses business data from the back-end system or service. Because the data on welcome cards is static, choosing a back end for those cards is optional. |
| Template Source     | Select the preferred source for the card definition:  
  ○ Template Manager – use a predefined set of card resources to create your cards.  
  ○ Zip – upload a zip file containing a set of resources that you can use to define your cards. |
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTML Template</td>
<td>Select from the following HTML templates:</td>
</tr>
<tr>
<td></td>
<td>○ Sample Template HR</td>
</tr>
<tr>
<td></td>
<td>○ Sample Template CRM</td>
</tr>
<tr>
<td></td>
<td>○ Sample Template Product</td>
</tr>
<tr>
<td></td>
<td>○ Sample Template Sales Orders</td>
</tr>
<tr>
<td></td>
<td>○ Sample Welcome</td>
</tr>
<tr>
<td>Resource Package</td>
<td>Select a zip file to upload.</td>
</tr>
<tr>
<td>Card Type</td>
<td>Select one of the following card types:</td>
</tr>
<tr>
<td></td>
<td>○ Default - these cards are created only when an explicit call is made to create them. For example, default cards are created when a third-party website places a request.</td>
</tr>
<tr>
<td></td>
<td>○ Welcome Card - custom cards that are downloaded automatically after registration.</td>
</tr>
<tr>
<td></td>
<td>○ Static cards - cards that are generated with default content.</td>
</tr>
<tr>
<td></td>
<td>○ Dynamic cards - you can add content to the cards. For example, you can add a welcome message to the card.</td>
</tr>
<tr>
<td></td>
<td>○ Server Managed Card - similar to welcome cards, but these cards are not automatically subscribed, the end-users need to explicitly subscribe to such cards.</td>
</tr>
<tr>
<td></td>
<td>○ Automatic Web Page Matching - allows the SAP Content to Go plugin installed in the Fiori launchpad to automatically detect applications for which cards can be created. This option can also be used by compatible third-party websites. Automatic web page matching is not supported for welcome cards and automatic instance generation cards.</td>
</tr>
<tr>
<td></td>
<td>○ Automatic Instance Generation - specify an OData query that returns an entity set, which automatically creates card instances. This option helps the user manage a set of instances automatically.</td>
</tr>
<tr>
<td>Description</td>
<td>Specify a description of the card in the default language.</td>
</tr>
<tr>
<td>New Card Notification</td>
<td>Create a notification message to be sent when a new card is added.</td>
</tr>
<tr>
<td>Changed Card Notification</td>
<td>Create a notification message to be sent when an existing card is updated.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Define parameters that can be used in action URLs. For example, you can define the value of comment parameter here, and the value will be substituted at runtime in the following action URL:</td>
</tr>
</tbody>
</table>
|                     | /sap/fiori/crossfndfioriinbox/sap/opu/odata/IWPGW/TASKPROCESSING;mo;v=2/Decision?SAP__Origin='EC3_800_BWF'&InstanceID=$
|                     | {ID2}&DecisionKey='0001'&Comments='${comment}'                                                                                         |
|                     | You can specify either a static value or a JSON path reference.                                                                        |

4. To use the selected HTML template as a welcome card:
   a. Select Welcome Card.
   b. To create a dynamic welcome card, enter the Data Endpoint URL, which specifies a JSON REST service from which data can be retrieved and bound into the card dynamically. For example, /sap/opu/odata/sap/EPMRA_PROD_MAN/Products. If you do not specify the Data Endpoint URL, a static welcome card is generated.
5. (Optional) Select **Enable Automatic Web Page Matching**.
   a. Enter a **Web Page Regular Expression** to identify a resource. For example, `Material-displayProductMaster/C_Product(ActiveProduct='{$1}')`, where the variable `{$1}` represents the unique product identifier (this identifier constructs the REST API query specified in the **Data Endpoint URL**) which is replaced with the actual value during runtime.
   b. Enter a **Data Endpoint URL**, which specifies a REST or OData service from which data can be retrieved. For example, `/sap/opu/odata/sap/EPMRA_PROD_MAN/Products('{$1}')`. 

   **i Note**
   If you are using automatic web page matching, **Web Page Regular Expression** and **Data Endpoint URL** are required.

6. (Optional) Select **Automatic Instance Generation**.
   a. Enter a **Query**. The server executes the specified query and automatically creates new cards.
   b. Select **Expect Collection** to get a response as JSON data or OData service via a collection-based query rather than an individual query request.
   c. Select a **Query Refresh Mode**, to specify how often the server executes the query.
      - Keep up-to-date – queries are automatically executed and are always up-to-date.
      - Use daily interval – queries are executed once daily at the specified time.
      - Use custom interval – queries are executed based on the data refresh time you specify.

   **i Note**
   You can now define parameters such as `{userId}`, `{language}`, and `{xsddatetime}` in the data endpoint URL for welcome cards, automatic instance generation, and automatic web page matching. The parameters are replaced with actual values at runtime, enabling users to obtain localized or customized cards.

7. Select **Sample Data** to preview the card to be delivered. You can either upload a JSON file or paste a JSON string to add data to the template.

8. Select **Images** to add, view, or delete images. For example, add an image and refer it in the HTML code to preview the image.

9. Select **Editor** to make changes to HTML or CSS code.
   - Select **Locale** to use a localized version of the content. You can now either select English or German.

   **i Note**
   You can now add JavaScript code in the HTML editor to customize cards. Use D3 JavaScript to incorporate graphics in your cards.

10. Select **Data Mapping** to map the handlebar bindings to an actual JSON path reference.

   **i Note**
   You can now bind JSON date values into the final HTML, using a suitable date format.

11. Select **Export** to export your card type template.
1.4.3.6.5.2 Defining Parameters

Define parameters to be used in action URLs.

Context

You can now define parameters for SAP Content to Go card templates. The value you specify for a parameter can be either a static value or a JSON path reference.

When you specify a static value for the parameter, the parameter value is directly substituted in the action URL at runtime. For example, if you specify static content for comment parameter, this value will replace $\text{comment}$ in the following action URL during runtime:

```
/sap/fiori/crossfndfioriinbox/sap/opu/odata/IWPGW/TASKPROCESSING;mo;v=2/
Decision?SAP__Origin='EC3_800_BWF'&InstanceID=\{$ID2\}&DecisionKey='0001'&Comments=\{$comment\}'
```

When you specify a JSON path reference, the server retrieves the value of the parameter from the JSON response and substitutes the value in the action URL at runtime. For example, to reference the value of someproperty from the following JSON response:

```
{"d": {"__metadata": {}, "someproperty": "some value"}}
```

You must first specify the JSON path reference as $d$.someproperty in the parameter definition.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications SAP Content to Go
2. Select an application to which you want to add parameters.
3. Define the parameters.
1.4.3.6.5.3 Editing a Card Type

Edit SAP Card Types from the Mobile Service for Development and Operations cockpit.

Procedure

1. In the Mobile Service for Development and Operations cockpit, select Mobile Applications SAP Content to Go.
2. Choose the card type you want to edit, and select Edit. Update the details as required.

1.4.3.6.5.4 Deleting a Card Type

Delete SAP Card Types from the Mobile Service for Development and Operations cockpit.

Procedure

1. In the Mobile Service for Development and Operations cockpit, select Mobile Applications SAP Content to Go.
2. Choose the card type you want to delete and select Delete.

Note

All card instances of the selected card type will be detected and you need to confirm if all instances need to be deleted.
1.4.3.6.5.5 Defining Categories

Create a category and define actions for card types. Users can execute the defined actions.

Context

Categories let the admin define the actions associated with card types. Actions then help the users determine how to proceed with a notification or with a card within the app. For example, if accept and reject operations are defined for a card type, when the user gets notified about the new card type, he or she can either accept or reject it.

Procedure

1. In the Mobile Service for Development and Operations cockpit, select [Mobile Applications] [SAP Content to Go]
2. Select Push Notification under Assigned Features (you may need to add it first). For more information on adding the feature, see Managing Application Features [page 45].
3. Select + to define the new category. Enter the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>Unique category name. For example, Accept_Reject.</td>
</tr>
<tr>
<td>Action</td>
<td>Unique action name. For example, Accept.</td>
</tr>
<tr>
<td>Note</td>
<td>You can define multiple actions for each category.</td>
</tr>
<tr>
<td>Localized title EN</td>
<td>Localized English title, which appears on the device.</td>
</tr>
<tr>
<td>Localized title DE</td>
<td>Localized German title, which appears on the device.</td>
</tr>
</tbody>
</table>

4. Click Save.
1.4.3.6.5.6 Defining Actions

Define an operation to be performed on the back end when the user executes an action.

Procedure

1. In the Mobile Service for Development and Operations cockpit, select Mobile Applications » SAP Content to Go.
2. Select an application, and then select Actions.
3. Select a category and provide the following information:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BehaviourAfterAction</td>
<td>Determines the status of the card once the action is successfully performed.</td>
</tr>
<tr>
<td></td>
<td>○ ACTIVE – perform the action and leave the card for the client to perform the action again.</td>
</tr>
<tr>
<td></td>
<td>○ INACTIVE – perform the action, then disable it, but do not remove the card.</td>
</tr>
<tr>
<td></td>
<td>○ DELETE – remove the card once the action has been performed.</td>
</tr>
<tr>
<td>XCSRFTokenUrl</td>
<td>Specify a URL from which to extract a XCSRF token. For example, a POST operation on the back-end system requires you to perform a GET, and the GET operation retrieves a XCSRF token. The URL refers to the service document root within which the action URL is defined.</td>
</tr>
<tr>
<td></td>
<td>Note: If a XCSRF token is required, set the following values for the action headers:</td>
</tr>
<tr>
<td></td>
<td>○ HeaderName – X-Requested-With</td>
</tr>
<tr>
<td></td>
<td>○ HeaderValue – XMLHttpRequestAction</td>
</tr>
<tr>
<td></td>
<td>For example, /sap/fiori/crossfndfioriinbox/sap/opu/odata/IWPGW/TASKPROCESSING;mo;v=2/.</td>
</tr>
<tr>
<td>ActionUrl</td>
<td>Specify the URL to be invoked on the back-end system to execute the operation. For example, /sap/fiori/crossfndfioriinbox/sap/opu/odata/IWPGW/TASKPROCESSING;mo;v=2/Decision?SAP__Origin='EC3_800_BWF'&amp;InstanceID=${ID2}&amp;DecisionKey='0001'.</td>
</tr>
<tr>
<td>HttpMethod</td>
<td>Specify the HTTP method. For example, GET, POST etc.</td>
</tr>
<tr>
<td>Field</td>
<td>Value</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>ActionBody</td>
<td>(Optional) Data that is supplied to execute the action. A POST method can use an ActionBody to create or update back-end content. For example, {&quot;some&quot;: &quot;jason&quot;}.</td>
</tr>
<tr>
<td>ActionHeaders</td>
<td>(Optional) Custom HTTP headers that can be sent with the URL.</td>
</tr>
</tbody>
</table>
| Action Parameters | Allows user to input parameter values while executing an action. For example, if a card is being approved or rejected by the user, you can prompt the user to specify a reason for the action performed.  
  ○ Name – property name.  
  ○ Label (EN) – title that appears on the card.  
  ○ Data Type – the input parameters data type, such as Edm.String, or Edm.DateTime.  
  ○ Maximum Length – maximum length of the input parameter.  
  ○ Is Key – whether the property is an OData key.  
  ○ Is Nullable – whether the property is nullable, typically true or false. |

4. Click Save.

Related Information

Manage SAP Content to Go Card Types [page 144]
Defining Categories [page 150]

1.4.3.6.5.7 Using iOS Spotlight Search

Use iOS Spotlight Search to find SAP Content to Go cards without launching the app.

Context

This feature allows users to use the Spotlight search field. For example, a search for "customer" returns every card containing that term is included in the search result.

Tapping a card that's part of the search results launches SAP Content to Go and shows the entire card.

i Note

Enabling Spotlight Search allows SAP Content to Go data to be shared with the iOS operating system. Disable this feature if you do not want your business data to be shared with iOS.
Procedure

1. In the Mobile Service for Development and Operations cockpit, select Mobile Applications > SAP Content to Go.
2. Select Client Policies feature.
3. Under Feature Restriction Policies, select Allowed to enable Spotlight. Unselect it to disable Spotlight.

Related Information

Manage SAP Content to Go Card Types [page 144]

1.4.3.6.6 Managing the Agentry Server Instance

The Agentry tile is available for accounts that have the Agentry capability enabled. Use the Mobile Services cockpit to manage Agentry features that are linked to your account.

- Provisioning an Agentry Account [page 153]
  - Provision Agentry accounts from the cockpit.
- Upgrading the Agentry Server [page 154]
  - You can upgrade your development account to the latest Agentry server version.
- Restarting the Agentry Server [page 154]
  - Restart the Agentry server for your account.

1.4.3.6.6.1 Provisioning an Agentry Account

Provision Agentry accounts from the cockpit.

Context

To set up an Agentry customer account, see Set Up Customer Accounts for Agentry [page 27]

If your account hasn’t been provisioned, choose an account type which needs to be provisioned. In the production landscape only Test and Production accounts can be provisioned. In the preview landscape only Development and Preview accounts can be provisioned.
Procedure

1. In the cockpit, select **Settings** ▶ **Agentry**.
2. Select **Provision**, which is available only if the account hasn't yet been provisioned. Once you've provisioned the account, **Restart** and **Upgrade** options also become available.
3. Enter user name and password. Select the landscape.
4. Select **Proceed**.
5. (Optional) Select **Get Current Status** to see how the provisioning is progressing.

1.4.3.6.6.2 Upgrading the Agentry Server

You can upgrade your development account to the latest Agentry server version.

Context

For development accounts an **Upgrade** button is enabled if a newer version of the Agentry server is available. With this, you can upgrade your development Agentry instance to the latest version. For other Agentry server instances such as preview, test, and production, the operations team upgrades the Agentry server instance to the latest version when they are released.

Procedure

1. In Mobile Service for Development and Operations cockpit, select **Settings** ▶ **Agentry**.
2. Select **Upgrade**.
3. Enter user name and password. Select **Proceed**.
4. (Optional) Select **Get Current Status** to get the status of the Upgrade action.

1.4.3.6.6.3 Restarting the Agentry Server

Restart the Agentry server for your account.

Context

All Agentry server instances include a **Restart** button. With this, you can restart the Agentry server instance.
Procedure

1. In Mobile Service for Development and Operations cockpit, select Settings > Agentry.
2. Select Restart.
3. Enter user name and password. Select Proceed.
4. (Optional) Select Get Current Status to get the status of the Restart action.

1.4.3.6.7 Sending a Push Notification

Send a push notification to all users of a push-enabled application, or a subgroup of recipients (such as all iOS devices in a particular time zone).

Context

This feature uses the existing push notification mechanism, but enables you to create messages to send to recipients. You can send a native push notification or an SMS text message notification. The feature requires you have the Notification User role to send push notifications (if the Send Push Notification button is disabled, the role needs to be added to your profile). Built in security ensures that you must wait a minute between sending notifications, to prevent spamming users. Even if you click Cancel to stop a message, you must wait the full minute before sending the next one.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid.
2. Select an application, and then select User Registrations to see a list of registered users.

You can click the Customize Table Columns icon to change the columns that appear.

Table 53: Registered Users Default Columns

<table>
<thead>
<tr>
<th>Category</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registration ID</td>
<td>The registration identifier of each application connection instance.</td>
</tr>
<tr>
<td>User Name</td>
<td>User name of the registered user.</td>
</tr>
<tr>
<td>Device Type</td>
<td>Device type for the registered user, such as Android, iOS, or Windows.</td>
</tr>
<tr>
<td>Category</td>
<td>Description</td>
</tr>
<tr>
<td>----------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Beta Tester</td>
<td>Indicates whether the user is a tester, versus a production user.</td>
</tr>
<tr>
<td>Last Connection</td>
<td>Identifies the last time the user connected.</td>
</tr>
<tr>
<td>Usage Upload</td>
<td>Whether the consent is granted or revoked. An accept icon and a date-time stamp with green color indicates when permission was granted. A decline icon and a date-time stamp with a red color indicates when permission was revoked. Blank indicates consent has not been granted by the user.</td>
</tr>
<tr>
<td>Actions</td>
<td>Indicates any actions you can take for the registered user, such as deleting the registration, or the current log settings.</td>
</tr>
</tbody>
</table>

3. Under **Registered Users**, select **Registration ID** to select all registered users, or select a subset of registered users.

You can use the filters to help find a subset of registered users. Also, for some fields, a list of suggestions appears as you type, based on your input. Select the one you want rather than continue typing.

### Table 54: Filter Criteria

<table>
<thead>
<tr>
<th>Filter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Name</td>
<td>Enter a specific user name, or the start of a user name.</td>
</tr>
<tr>
<td>User Locale</td>
<td>Enter the language. This field is not validated, so you can enter any string value. Use the field if you know one or more values that users entered during registration, such as EN or German.</td>
</tr>
<tr>
<td>Time Zone</td>
<td>Enter one or more time zones, separated by commas. This field is not validated, so you can enter any string value. Use the field if you know one or more values that users entered during registration, such as UTC-1 or PST.</td>
</tr>
<tr>
<td>Device Type</td>
<td>Select one or more device type, such as iOS and Android.</td>
</tr>
<tr>
<td>Push Group</td>
<td>Enter a push group name. This field is not validated, so you can enter any string value. Use the field to create an ad hoc group for filtering. Push groups do not persist.</td>
</tr>
<tr>
<td>Filter</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
</tbody>
</table>
| Email                  | Enter one or more email address separated with the vertical bar ( | ). Wildcards are permitted. Valid formats include:  
  ○ *.com  
  ○ abc*@  
  ○ abc@company.fr |
| Registration Timeframe | Select a time frame from the list: Yesterday, Last 7 Days, Last 4 Weeks, Last 3 Months, Last 6 Months, Last 12 Months, or Custom Defined (use the date-time picker to define a range). |

4. Click **Send Push Notification**.

**i Note**
The Send Push Notification option only appears for push-enabled applications, or applications with SAP’s default push configuration enabled. Also, you are notified if the application registration is not push notification enabled.

5. In the **Send Push Notification** dialog, identify the primary and fallback push channels for sending the push notification, and compose the message.
   - **Primary Channel** – identify the primary push channel to use for this notification:
     - Push Notification (Native)
     - Text Message (SMS)
   - **Fallback Channel** – if you selected native as the primary push channel, you can select Text Message (SMS) as the fallback push channel for users who have agreed to receiving text messages. If you selected SMS as the primary push channel, you can select push as the fallback if the selected registrations include native push enabled users.
   - **Message** – type the message to send to registered application users that meet your search criteria.
     For example, *See the new version of our widget!*.

**i Note**
Since many service providers charge for SMS notifications, a good strategy may be to use native as the primary channel, and SMS as the fallback. This helps ensure that as many selected users as possible receive the notification.

You are notified about the number of selected users that will receive the notification. You can adjust your strategy as needed.

6. Click **Send** and **OK** to confirm. Once the notification is sent, you are prevented from sending another notification for 60 seconds. Even if you click **Cancel** to stop a message, you must wait the full minute before sending the next one (a second timer appears briefly for the second message).

*The Notification has been sent successfully!* appears. Use the **Analytics** ▶ **Push Statistics** ▶ **Notifications (by Application ID)** report to follow up on native push and SMS message status.
1.4.3.6.8 **Client and Server Data Report**

You can use Mobile Service for Development and Operations cockpit to view aggregated data for native, hybrid, and Web applications and its clients. Data reports appear in graphical form and provide summaries of client and server sessions.

1.4.3.6.8.1 **Viewing Client Data Report**

View aggregated data for native, hybrid, and Web applications.

**Prerequisites**

Applications for which you are collecting data must be enabled to collect data.

**Context**

Data reports are summaries of user sessions, which you can filter by Application ID, Application Version, Operating System, Operating System Version, Device Model, or Time Frame. You can download reports to CSV files and import them into tools such as SAP Lumira or Excel.

Data is aggregated in real time, and you can view them for a subset of applications, which can be a powerful research and monitoring tool.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select `Analytics ➤ Client Data Report`.
2. To filter client statistics, select the appropriate `Data` and `Time Frame` options:
   - **User Sessions per Application, Time Period** – view the number of user sessions for each application during a time period.
   - **User Sessions by OS, Version, and Device Model** – view user sessions based on the operating system, version, and device model.
   - **OS Share of User Sessions** – view the number of user sessions shared between operating systems over a period of one month.
   - **OS, OS Version Share of User Sessions** – view the number of user sessions based on the operating systems and their versions over a period of one month.
   - **User Session Length (in milliseconds)** – view user sessions based on the session durations.
3. To further refine your search, select `Application ID, Application Version, Operating System, Operating System Version, or Device Model`; choose the appropriate options and click `Go`.
4. To change the data-presentation format, click the Table View icon.

5. To download a client data report to a CSV file, click the Download icon.

1.4.3.6.8.2 Viewing Server Data Report

View data for the mobile service for development and operations. Data reports appear in graphical form and provide a summary of application registrations, number of users and requests, and response times.

Prerequisites

Enable applications to collect data; to get registration information; there must be registered users.

Context

Data is aggregated in real time, which can be a powerful research and monitoring tool.

Procedure

2. Select the appropriate Data option to filter the usage reports:
   - Registrations (by Time) – view the number of active registrations during a time period.
   - Registrations (by Device Type) – view the number of registrations group by device types, for example, Android, BlackBerry, and iOS.
   - Users (by Application) – view the number of active users per application.
   - Requests – view the number of requests made during a time period. Does not include offline requests.
   - Response Time – view the response time, in milliseconds, per time period. Response time is reported for Mobile related processing time, authentication time, and the back-end response time. Does not include offline response times.
   - Offline Requests – view the number of offline requests made during a time period. Requests are reported only for offline OData applications. Operations requests include:
     - Initial Download – complete the initial build of the database file on the server, and download the offline data store.
     - Refresh – complete refresh of the offline data store from the server. May be full or delta-only.
     - Flush – complete flush of local changes from the offline data store to the server. Delta-only.
   - Offline Response Time – view the average response time for offline requests per time period. Response time is reported only for offline OData applications.
○ **Threshold Hit Count for Application** – view the number of times that incoming requests hit the threshold that is set for the application. Incoming requests include registration, online, and offline requests.

○ **Threshold Hit Count for Destination** – view the number of times that incoming requests hit the threshold that is set for the destination. Incoming requests includes online requests.

○ **Deprecated Version Requests** – view the number of requests per deprecated app versions during a particular time period.

○ **Outbound Traffic (by Application)** – view the outbound HTTP traffic statistics for one or more applications.

○ **Outbound Traffic (by Destination)** – view the outbound HTTP traffic statistics for the selected destination.

3. Select the **Time Frame** to gather and display the statistics report: Today, Yesterday, Last 7 Days, Last 4 Weeks, Last 3 Months, or Last 12 Months.

4. To change the format in which the data appears, select the appropriate icon in the top-right corner of the page.

   For offline request and response time statistics, you can use the legend to drill down and view more information about the selected statistics, including time period, category, number of milliseconds, and number of values represented.

   ○ You can download the report in .CSV format.

   ○ You can zoom in or out the graphic, if it is too large to view on the current window size.

   For threshold hit counts, you can use the legend to drill down and view more information about the selected statistics. You can also view the report data in a table format. Keep in mind:

   ○ If you select multiple applications, the value reflects a sum value.

   ○ If you select a single application or connection, the value reflects only the selected application or connection.

   ○ If the threshold hit count chart shows no data, it indicates that the server is in a good state. The actual request count is lower than the threshold value.

   ○ If the charts show a lot of data, the server load may be too high, or the threshold value may be set too low. You may want to adjust the thresholds to better control incoming traffic requests.

   For outbound traffic statistics, you can use the legend to drill down and view more information about the selected statistics. You can also view the report data in a table format. These values are for statistical comparison only, and are not comparable to values used for billing.

5. To filter the data in the report, select the appropriate filters on the top-right corner of the page.

   You can filter by Application ID, by Vendor, by Destination (For example, if you select **Threshold Hit Count for Destination** and **Outbound Traffic by Destination**), by Version (For example, if you select **Deprecated Version Requests**).

   a. Select the items from the drop-down list.

   b. Click **Go**.

   c. To reset the filters, select **Reset Filter**.
1.4.3.6.8.3 Viewing Push Statistics

View push notification statistics for push enabled apps from Push Statistics. You can see notification details, notifications by applications, and notifications by operating system.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics Push Statistics.
2. Select the appropriate Usage option to filter the push notification report:
   - Notifications in Detail – view push notification counts by time period, application, and operating system. Count categories include notifications sent, received, resulting in error, and consumed successfully.
   - Notifications (by Application ID) – view push notification counts by time period and application. Count categories include notifications sent, received, resulting in error, and consumed successfully.
   - Notifications (by OS) – view push notification counts by time period and operating system. Count categories include notifications sent, received, resulting in error, and consumed successfully.
3. Select the Time Frame to gather and display the statistics report. The time frame selected determines how the time format appears:
   - Today (default) and Yesterday – the time format is <hh - hh>. Twenty four hours is divided into six parts for comparison: 00 - 04; 04 - 08; 08 -12; 12 - 16; 16 - 20; and 20 - 24.
   - Last 7 Days – the time format is <yyyy/MM/dd>; for example, 2016/11/30.
   - Last 4 Weeks – the time format is <yyyy/Www>, where <Www> is the week of the year; for example, W22 is week 22 in the year.
   - Last 3 Months and Last 12 Months – the time format is <yyyy/MM>; for example, 2016/12
4. To change the format in which the data appears, select the appropriate icon in the top-right corner of the page. You can view the report in table format, and download the report in .CSV format. You can click Reset to reset the filters. When the legend appears, you can use it to drill down and view more information about the selected statistics.
5. To filter the data in the report, select the appropriate filters on the top-right corner of the page.
   a. Select the items from the drop-down list.
   b. Click Go.
   c. To reset the filters, select Reset Filter.

1.4.3.6.9 Application Logs and Trace Files

Set the verbosity for application and component logging, and define how long to keep logs and trace files. You can view all application logs or a subset of your choice, and drill down to view detailed log and trace information if available.

Setting Log Levels [page 162]
You can change the logging level for one or more logging components. In a troubleshooting situation, you may want to increase the log level to capture more details.

**Enabling Application Traces [page 163]**
Enable application traces for selected SAP Cloud Platform mobile service for development and operations logging components. Application tracing captures additional business data for a request (such as message data, HTTP headers, and URIs), which you can use to troubleshoot application problems. Enable tracing for individual logging components on an as-needed basis.

**Viewing Event Logs [page 165]**
Use log information to troubleshoot application problems. Use search criteria to specific log records and statements. Depending on what information is captured, log information can include entries from event logs.

**Viewing Technical Logs [page 167]**
Use log information to troubleshoot application problems. Use search criteria to find specific log records and statements needed to diagnose a problem.

**Purging Logs and Trace Files [page 169]**
Set up a schedule for purging log and trace files. You can also purge them immediately, on demand.

### 1.4.3.6.9.1 Setting Log Levels

You can change the logging level for one or more logging components. In a troubleshooting situation, you may want to increase the log level to capture more details.

**Context**

**Note**
Logging detailed information consumes system resources, so SAP recommends that you change the log level only when you suspect a serious problem, or are testing a theory. In most situations, it is sufficient to log only errors and warnings.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select Settings ➤ Log Settings.
2. For each component, select a log level.
Table 55: System Logging Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin</td>
<td>Logs system messages that are related to SAP Cloud Platform mobile service for development and operations administration.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Logs system messages that are related to all HTTP connections.</td>
</tr>
<tr>
<td>Content to Go</td>
<td>Logs system messages that are related to Content to Go.</td>
</tr>
<tr>
<td>Foundation</td>
<td>Logs system messages for core SAP Cloud Platform mobile service for develop­ment and operations functionality.</td>
</tr>
<tr>
<td>Hybrid Application Management</td>
<td>Logs system messages that relate to managing hybrid apps through Mobile Serv­ice for Development and Operations cockpit or an API, and client interactions for requesting and downloading updated hybrid apps.</td>
</tr>
<tr>
<td>Offline</td>
<td>Logs system messages that are related to the offline OData service.</td>
</tr>
<tr>
<td>Proxy</td>
<td>Logs system messages that are related to any client-back end interactions using SAP Cloud Platform mobile service for development and operations as a proxy (for example, OData requests).</td>
</tr>
<tr>
<td>Push</td>
<td>Logs system messages that are related to push actions.</td>
</tr>
<tr>
<td>Registration</td>
<td>Logs system messages that are related to application registration.</td>
</tr>
<tr>
<td>Security</td>
<td>Logs system messages that are related to security.</td>
</tr>
<tr>
<td>Statistics</td>
<td>Logs system messages that are related to usage statistics.</td>
</tr>
</tbody>
</table>

Table 56: Logging Levels

<table>
<thead>
<tr>
<th>Log Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug</td>
<td>For debugging purposes, includes extensive and low-level information.</td>
</tr>
<tr>
<td>Info</td>
<td>Informational text, used mostly for echoing what has been performed.</td>
</tr>
<tr>
<td>Warn</td>
<td>The application can recover from the anomaly, and fulfill the task, but requires at­tention from the developer or operator.</td>
</tr>
<tr>
<td>Error</td>
<td>The application can recover from the error, but cannot fulfill the task due to the error.</td>
</tr>
</tbody>
</table>

1.4.3.6.9.2 Enabling Application Traces

Enable application traces for selected SAP Cloud Platform mobile service for development and operations logging components. Application tracing captures additional business data for a request (such as message data, HTTP headers, and URIs), which you can use to troubleshoot application problems. Enable tracing for individual logging components on an as-needed basis.
Context

Note
Tracing can impact server performance. Enable traces only when required for debugging or user support.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Settings ➤ Log Settings.
2. Enable tracing as required for each logging component.

Table 57: System Logging Components

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin</td>
<td>Logs system messages that are related to SAP Cloud Platform mobile service for development and operations administration.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Logs system messages that are related to all HTTP connections.</td>
</tr>
<tr>
<td>Content to Go</td>
<td>Logs system messages that are related to Content to Go.</td>
</tr>
<tr>
<td>Foundation</td>
<td>Logs system messages for core SAP Cloud Platform mobile service for development and operations functionality.</td>
</tr>
<tr>
<td>Hybrid Application Management</td>
<td>Logs system messages that relate to managing hybrid apps through Mobile Service for Development and Operations cockpit or an API, and client interactions for requesting and downloading updated hybrid apps.</td>
</tr>
<tr>
<td>Offline</td>
<td>Logs system messages that are related to the offline OData service.</td>
</tr>
<tr>
<td>Proxy</td>
<td>Logs system messages that are related to any client-back end interactions using SAP Cloud Platform mobile service for development and operations as a proxy (for example, OData requests).</td>
</tr>
<tr>
<td>Push</td>
<td>Logs system messages that are related to push actions.</td>
</tr>
<tr>
<td>Registration</td>
<td>Logs system messages that are related to application registration.</td>
</tr>
<tr>
<td>Security</td>
<td>Logs system messages that are related to security.</td>
</tr>
<tr>
<td>Statistics</td>
<td>Logs system messages that are related to usage statistics.</td>
</tr>
</tbody>
</table>
1.4.3.6.9.3 Viewing Event Logs

Use log information to troubleshoot application problems. Use search criteria to specific log records and statements. Depending on what information is captured, log information can include entries from event logs.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics Logs.
2. Select Event Logs.
3. (Optional) Enter log search criteria.
   - To view event logs for a specific application, enter values for one or more of the following optional filters:
     - Application ID
     - Level (Optional).
     - Correlation ID (Optional).
     - Component (Optional).
     - Time Frame (Optional).
     - User name (Optional).
     and click Go, then select the application.
   - To view event log messages for a specific level, select Level, then select the logging level. Leave blank to include all levels.

   **Note**

<table>
<thead>
<tr>
<th>Table 58: Logging Levels</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Log Level</strong></td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Debug</td>
</tr>
<tr>
<td>Info</td>
</tr>
<tr>
<td>Warn</td>
</tr>
<tr>
<td>Error</td>
</tr>
</tbody>
</table>

   - To view event log messages for a specific component, select Component, then select the component; to select all components, leave this field blank. Select Go.

<table>
<thead>
<tr>
<th>Table 59: System Logging Components</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Component</strong></td>
</tr>
<tr>
<td>----------------</td>
</tr>
<tr>
<td>Admin</td>
</tr>
<tr>
<td>Component</td>
</tr>
<tr>
<td>-------------------------------</td>
</tr>
<tr>
<td>Connectivity</td>
</tr>
<tr>
<td>Content to Go</td>
</tr>
</tbody>
</table>
| Foundation                    | Logs system messages for core SAP Cloud Platform mobile service for develop-
|                               | ment and operations functionality.                                          |
| Hybrid Application Management | Logs system messages that relate to managing hybrid apps through Mobile     |
|                               | Service for Development and Operations cockpit or an API, and client interac-
|                               | tions for requesting and downloading updated hybrid apps.                   |
| Offline                       | Logs system messages that are related to the offline OData service.         |
| Proxy                         | Logs system messages that are related to any client-back end interactions us-
|                               | using SAP Cloud Platform mobile service for development and operations as a  |
|                               | proxy (for example, OData requests).                                       |
| Push                          | Logs system messages that are related to push actions.                      |
| Registration                  | Logs system messages that are related to application registration.          |
| Security                      | Logs system messages that are related to security.                         |
| Statistics                    | Logs system messages that are related to usage statistics.                  |

- To limit the results within a time frame, select the time period; or set the time manually by selecting **Custom Defined**, entering a date range, and selecting **OK**.
- To view log messages for requests initiated by a specific user, enter a user name.

Logging information is based on your search criteria.

Table 60:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>The time and date stamp of the log entry.</td>
</tr>
<tr>
<td>Level</td>
<td>Level value, typically <strong>ERROR</strong> or <strong>WARN</strong> or <strong>INFO</strong> or <strong>DEBUG</strong>.</td>
</tr>
<tr>
<td>Component</td>
<td>Component values are Admin, Connectivity, Content to Go, Foundation, Hybrid</td>
</tr>
</tbody>
</table>
|                | Application Management, Offline, Proxy, Push, Registration, Security, and Sta-
|                | tistics.                                                                    |
| User Name      | The name of the user associated with the application ID.                    |
| Mesage         | The link to detailed log information associated with the execution request. |

4. (Optional) To view event log messages, select a row, and click the **icon**.
5. (Optional) To download a text version of the event log file to the **Downloads** directory, select one or more rows, and click the **Download** icon.
1.4.3.6.9.4 Viewing Technical Logs

Use log information to troubleshoot application problems. Use search criteria to find specific log records and statements needed to diagnose a problem.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics → Logs.
2. Select Technical Logs.
3. Enter log search criteria:
   - To view technical logs for a specific application, enter the values for the following filters in your search criteria:
     - Application ID (Optional).
     - Level (Optional).
     - Correlation ID (Optional).
     - Component (Optional).
     - Type (Optional).
     - Time Frame (Optional).
     - User name (Optional).
     and click Go, then select the application.
   - To view technical log messages for a specific level, select Level, then select the logging level.

<table>
<thead>
<tr>
<th>Log Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Debug</td>
<td>For debugging purposes, includes extensive and low-level information.</td>
</tr>
<tr>
<td>Info</td>
<td>Informational text, used mostly for echoing what has been performed.</td>
</tr>
<tr>
<td>Warn</td>
<td>The application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator.</td>
</tr>
<tr>
<td>Error</td>
<td>The application can recover from the error, but cannot fulfill the task due to the error.</td>
</tr>
</tbody>
</table>

   - To view technical log messages for a specific request type, select the request Type such as Application Settings, Client Log, Deregistration and so on; or you can select All, and click Go.
   - To view technical log messages for a specific component, select Component; for all components, select all or leave it blank, and click Go.

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Admin</td>
<td>Logs system messages that are related to SAP Cloud Platform mobile service for development and operations administration.</td>
</tr>
<tr>
<td>Connectivity</td>
<td>Logs system messages that are related to all HTTP connections.</td>
</tr>
<tr>
<td>Component</td>
<td>Description</td>
</tr>
<tr>
<td>-----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Content to Go</td>
<td>Logs system messages that are related to Content to Go.</td>
</tr>
<tr>
<td>Foundation</td>
<td>Logs system messages for core SAP Cloud Platform mobile service for develop-</td>
</tr>
<tr>
<td></td>
<td>ment and operations functionality.</td>
</tr>
<tr>
<td>Hybrid Application Management</td>
<td>Logs system messages that relate to managing hybrid apps through Mobile</td>
</tr>
<tr>
<td></td>
<td>Service for Development and Operations cockpit or an API, and client inter-</td>
</tr>
<tr>
<td></td>
<td>actions for requesting and downloading updated hybrid apps.</td>
</tr>
<tr>
<td>Offline</td>
<td>Logs system messages that are related to the offline OData service.</td>
</tr>
<tr>
<td>Proxy</td>
<td>Logs system messages that are related to any client-back end interactions</td>
</tr>
<tr>
<td></td>
<td>using SAP Cloud Platform mobile service for development and operations as a</td>
</tr>
<tr>
<td></td>
<td>proxy (for example, OData requests).</td>
</tr>
<tr>
<td>Push</td>
<td>Logs system messages that are related to push actions.</td>
</tr>
<tr>
<td>Registration</td>
<td>Logs system messages that are related to application registration.</td>
</tr>
<tr>
<td>Security</td>
<td>Logs system messages that are related to security.</td>
</tr>
<tr>
<td>Statistics</td>
<td>Logs system messages that are related to usage statistics.</td>
</tr>
</tbody>
</table>

- To limit the results within a time frame, select the time-period; or you can also set the time manually by selecting Custom Defined, then choose From and To dates and times respectively, and click OK.
- To view technical log messages for requests initiated by a specific user, enter the User Name.

Logging information is based on your search criteria.

Table 63:

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>The time and date stamp for the log entry.</td>
</tr>
<tr>
<td>Level</td>
<td>Level value, typically ERROR or WARN or INFO or DEBUG.</td>
</tr>
<tr>
<td>Registration ID</td>
<td>The unique connection identifier that makes the request to the server.</td>
</tr>
<tr>
<td>User Name</td>
<td>The name of the user associated with the application ID.</td>
</tr>
<tr>
<td>Component</td>
<td>The component type, such as connectivity, registration, offline, and so forth.</td>
</tr>
<tr>
<td>Type</td>
<td>The request type, such as application settings, client log deregistration, hybrid, notification, and so forth.</td>
</tr>
<tr>
<td>Application 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 application ID is used for registration and client requests.</td>
</tr>
</tbody>
</table>

4. (Optional) To view technical log messages, select a row, and click the icon.
5. (Optional) To download a text version of the log file to the Downloads directory, select one or more rows, and click the Download icon.
6. (Optional) To view technical log messages for multiple rows within the browser window, select the rows and click the View icon.
You can use a browser to search through the information using keywords. You can copy the content in the Log Details window to a text editor.

1.4.3.6.9.5 Purging Logs and Trace Files

Set up a schedule for purging log and trace files. You can also purge them immediately, on demand. (??)

Context

By default, logs are purged when the database reaches the size you specify, for example, the following command sets the threshold at 80, so logs are purged when the database is 80% full:

-Dcom.sap.mobile.platform.server.threshold.logpurge.threshold=80

Procedure

1. In Mobile Service for Development and Operations cockpit, select Settings > Log Settings. You can see the existing components, log level details, and event log information.

2. Under Purge Settings:
   a. Select the number of days (1–30) to keep server error logs.
   b. Select the number of days (1–30) to keep server success logs.
   c. Select the number of days (1–30) to keep client error logs.
   d. Select the number of days (1–30) to keep client success logs.
   e. Select the number of days (1–30) to keep error trace logs.
   f. Select the number of days (1–30) to keep success trace logs.

3. To immediately delete all log and trace files from the database, click Purge Now.

1.4.3.6.10 Tracing Network Activity

Application users can trace network activity based on user name, connection, application, or content type. You can download tracing reports to either a .zip or a .har file.

Prerequisites

Configure the tracing properties before you turn tracing on.
Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics > Network Traces.

2. By default, SAP Cloud Platform mobile service for development and operations traces all users, connections, applications, and content types. To trace specific activity, enter values for one or more of the following properties:
   - User Name
   - Registration ID

   **i Note**
   If you leave the User Name field or Registration ID field as blank, then all the values are selected by default.

   - Application ID or select All.
   - Content Type or select All.

3. For Logging Limit, select either:
   - To limit the user to log only for 2KB of payload, select Limit to 2KB, or
   - To limit the user to network trace for only one hour, select Limit to one hour.

4. To start logging network trace, click Start Recording.

5. To stop logging network trace, click Stop Recording.

6. To download a tracing report:
   a. Filter the report by specifying one or more of the following property values:
      - User Name
      - Registration ID

      **i Note**
      If you leave the User Name field or Registration ID field as blank, then all the values are selected by default.

      - Application ID or select All.
      - Content Type or select All.
      - Time Frame
   b. Select the date range.
   c. Select the output file type, ZIP or HAR, and click the Download icon.

7. To purge the trace log, select Purge Now.
1.4.3.6.11 Viewing the User Audit Log

Use audit log information to view user actions over the most recent six months.

Context

The audit log provides an ongoing record of the sequence of operations, procedures, event activities, or communications initiated by individual users. By default, the audit log is always enabled.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Analytics Auditing.
2. (Optional) Enter log search criteria to narrow the focus to an area of interest, and then click Go.
   - **User** – specify a user.
   - **Source IP** – specify a source IP address.
   - **User Action** – select one or more user actions, such as Create, Update, Delete, Publish, or UnPublish.
   - **Object Type** – select one or more object types, such as Application, Connection, Account, SecureLoginServer, or ClientRegistration.
   - **Object Name** – specify an object.
   - **Status** – select one or more statuses, such as Success or Failure.
   - **Time Frame** – select a time frame for the search, such as Last Hour (default), Last 24 Hours, Last 7 Days, Last 4 Weeks, Last 3 Months, and Last 6 Months. Select Custom Defined to enter a specific date and time range.
3. View auditing information based on your search criteria. Optionally, you can sort auditing information by column, in ascending or descending order.

Table 64: Audit Log Properties

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time</td>
<td>The time of the log entry in the format YYYY-MM-DD hh:mm:ss.</td>
</tr>
<tr>
<td>User</td>
<td>The user who performed the logged action.</td>
</tr>
<tr>
<td>Source IP</td>
<td>The source IP address associated with the user action, in the format xxx.xxx.xxx.xxx.</td>
</tr>
<tr>
<td>User Action</td>
<td>The user action performed, such as Create, Update, Delete, Publish, or UnPublish.</td>
</tr>
</tbody>
</table>
### Properties and Description

<table>
<thead>
<tr>
<th>Properties</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object Type</td>
<td>The object type that was involved with the action, such as Application, Connection, Account, SecureLoginServer, or ClientRegistration.</td>
</tr>
<tr>
<td>Object Name</td>
<td>The name of the object that was involved with the action.</td>
</tr>
<tr>
<td>Status</td>
<td>The status of the action, either Success or Failure. Failure indicates a server validation error, for example, if a user tried to delete a connection that is used by another application.</td>
</tr>
</tbody>
</table>

4. (Optional) Click a log entry to view audit log details, including the revision history showing old and new values.

5. (Optional) Click *Export* to export the User Auditing log as a CSV report.
   - Use the date picker to specify the date and time range.
   - Click *Export*.
   - Open the downloaded file to view the data.

6. (Optional) Click *Reset* to clear your log search filters.

---

### 1.4.3.6.12 Manage Signing Profiles

You can create signing profiles for Android and iOS applications. As an administrator, you can control access to the signing profiles that are available for an account.

- **Creating a Signing Profile** [page 172]
  As an administrator, you can create signing profiles for Android and iOS applications.

- **Editing a Signing Profile** [page 173]
  As an administrator, you can edit an existing uploaded signing profile for Android or iOS platforms.

- **Deleting a Signing Profile** [page 174]
  As an administrator, you can delete an existing uploaded signing profile for Android or iOS platforms.

---

### 1.4.3.6.12.1 Creating a Signing Profile

As an administrator, you can create signing profiles for Android and iOS applications.

**Procedure**

1. Select *Settings > Signing Profiles*.
2. Select *Generate* to generate a new Android signing profile, and provide the following information:
Required information:
- Profile Name
- Validity (in years)
- Common Name (current user’s name)

Optional information includes certificate information, such as the Organization, Organization Unit, City or Locality, State or Province, and Country Code.

3. Select Upload and choose whether to upload an Android or an iOS signing profile, and provide the following information

   For the Android platform:
   - Profile Name
   - KeyStore File (in .keystore or .jks format)
   - KeyStore Password
   - KeyChain Alias

   For the iOS platform:
   - Profile Name
   - Signing Certificate (in .p12 format)
   - Private Key passphrase
   - Provisioning Profile (in .mobileprovisioning format)

4. Save the new signing profile.
   The certificate expiration date is stored and shown on the profile

Related Information

Manage Signing Profiles [page 172]

1.4.3.6.12.2 Editing a Signing Profile

As an administrator, you can edit an existing uploaded signing profile for Android or iOS platforms.

Procedure

1. Select Settings > Signing Profiles
2. Select the profile you want to edit and make the required changes. You cannot change the name of the signing profile.

   Note
   You cannot edit generated Android signing profiles.
3. Click Save.

Related Information

Manage Signing Profiles [page 172]

1.4.3.6.12.3 Deleting a Signing Profile

As an administrator, you can delete an existing uploaded signing profile for Android or iOS platforms.

Procedure

1. Select Settings → Signing Profiles.
2. Select the profile you want to delete.
3. Click Delete.
4. Click OK.

Related Information

Manage Signing Profiles [page 172]

1.4.3.6.13 Monitoring Hybrid Application Versions

When multiple versions of a hybrid application are running on registered devices, you can see the percentage of devices that are running each version of the application.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications → Native/Hybrid.
2. Select an application.
3. To monitor the hybrid application version, in the Info tab, select the + icon in the Assigned Features section.
4. In the Add feature window, select *App Update* and click *OK*.
5. Select *App Update*.
6. Select *Hybrid Application Versions*.
7. Select the *Registration ID*.
   For active versions of the selected application, a pie chart shows the percentage of registered devices on which each version is running.
8. To filter the report, enter a value for one of the following properties, and click the *Search* icon.
   ○ Registration ID
   ○ User Name
   ○ Device Type
   ○ Client Revision
9. To sort results based on one of these properties, select and select the search field. You can sort the results in ascending or descending order.
10. To customize the table view, click and select the columns.
11. To export the table data of a specific application version to a *CSV* file, select the row and click its *Download* icon.
12. To see more information about a specific application version, click the row.
13. To refresh the report, click the *Refresh* icon.

### 1.4.3.6.14 SAP License Auditing

The SAP Cloud Platform mobile service for development and operations License Audit feature enables you to generate an SAP audit measurement file in accordance with the SAP License Auditing process.

The SAP License Audit feature measures the total number of users registered with SAP Cloud Platform mobile service for development and operations, and the number of bytes sent in responses to registered users. The results are written to an audit XML file, which you can send to SAP according to the instructions in your SAP License Audit notice.

**Exporting Global Licensing Audit Services Reports [page 176]**
Export a Global Licensing Audit Services (GLAS) report as an XML file that you can send to SAP for uploading to the SAP Global Auditing License Service.

**SAP Application Users Tracked with SAP License Audit Overview [page 176]**
Review a list of application users that are tracked with the Global License Auditing Service. These application users are identities that are registered with SAP Cloud Platform mobile service for development and operations during automatic onboarding, or when a user is manually registered by an administrator.
1.4.3.6.14.1 Exporting Global Licensing Audit Services Reports

Export a Global Licensing Audit Services (GLAS) report as an XML file that you can send to SAP for uploading to the SAP Global Auditing License Service.

Context

GLAS reports include statistics for active users, such as the peak number of active users during a month. An active user is one who connects to SAP Cloud Platform mobile service for development and operations at least once during a three-month period, using a single application. If a user accesses the platform using multiple applications, they are counted as one active user for each application.

Procedure

2. Click License Report.
   An XML file that contains the auditing information is downloaded to your local hard drive.
3. When you are ready to share the auditing results, send the file to SAP as instructed in your SAP License Audit notice.

   Note

   For information about uploading the GLAS file to SAP Global Auditing License Service, see: https://support.sap.com/keys-systems-installations/measurement.html.

1.4.3.6.14.2 SAP Application Users Tracked with SAP License Audit Overview

Review a list of application users that are tracked with the Global License Auditing Service. These application users are identities that are registered with SAP Cloud Platform mobile service for development and operations during automatic onboarding, or when a user is manually registered by an administrator.

   Note

   Application users for applications created with SAP Cloud Platform mobile service for development and operations 1.1 or earlier are not included in the audit measurement results.
1.4.4 System Administration

At the system level, you can configure Cross-Origin Resource Sharing and the System for Cross-Domain Identity Management authentication.

**Configuring Cross-Origin Resource Sharing [page 177]**
To enable cross-origin resource sharing (CORS), configure header parameters for the system, which enables a request from one origin to access resources in another origin.

**Configuring SCIM Authentication [page 179]**
Delegate basic authentication requests to the System for Cross-Domain Identity Management (SCIM) server for either the SAP Cloud Platform tenant account or SAP Cloud Platform mobile service for development and operations applications only.

**Managing Inbound Request Traffic [page 180]**
Set thresholds for inbound registration requests, online requests, and offline requests. When the threshold is reached, requests are throttled and the server generates HTTP error code “429 Too Many Requests”.

**Configuring a Secure Login Server Profile [page 181]**
Configure a secure login server profile to enable applications to use a certificate to access trusted services.

1.4.4.1 Configuring Cross-Origin Resource Sharing

To enable cross-origin resource sharing (CORS), configure header parameters for the system, which enables a request from one origin to access resources in another origin.

**Prerequisites**

See:
- W3C Cross-Origin Resource Sharing
- Wikipedia Cross-Origin Resource Sharing
**Context**

A configuration identifies header components that must be included to allow resource sharing across origins. If a resource in one origin sends a valid "preflight" HTTP request that includes the required parameters, the resource can be granted access to resources in the second origin. If the required parameters are not included, cross-origin requests are not allowed.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select **Settings > Account Security**.
2. Configure the Cross Domain Access parameters:

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
<th>Default</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin</td>
<td>A comma-delimited list of URIs that can access the resource, for example, <a href="http://example.com">http://example.com</a> or http://<em>.example.com. The default value &quot;</em>&quot; means any URI can access the resource. An empty value prevents cross-origin resource sharing; only URIs in the same origin can access the resource.</td>
<td>*</td>
</tr>
<tr>
<td>Expose Headers</td>
<td>A comma-delimited list of response headers that browsers can access.</td>
<td>None</td>
</tr>
<tr>
<td>Max Age</td>
<td>The number of seconds for which the results of a request can be cached. The default is 3600 seconds (60 minutes).</td>
<td>3600</td>
</tr>
<tr>
<td>Allow Credentials</td>
<td>Always set to On. The server includes cookies when it submits requests.</td>
<td>On</td>
</tr>
<tr>
<td>Methods</td>
<td>A comma-delimited list of HTTP methods (such as GET and POST) that are allowed when accessing the resource.</td>
<td>None</td>
</tr>
<tr>
<td>Headers</td>
<td>A comma-delimited list of HTTP request headers that you can specify in requests.</td>
<td>None</td>
</tr>
</tbody>
</table>

**Note**

An empty value means any requested headers are accepted.

The server includes the following headers in its responses: accept, authorization, maxdataserviceversion, and x-smp-appcid.

3. Click **Save**. All entries must be valid.
1.4.4.2 Configuring SCIM Authentication

Delegate basic authentication requests to the System for Cross-Domain Identity Management (SCIM) server for either the SAP Cloud Platform tenant account or SAP Cloud Platform mobile service for development and operations applications only.

Context

If you are using Basic Authentication, the default setting "Default Identity Provider" delegates Basic authentication to the SAP ID service. You can configure the application to use global SCIM settings instead, or to override the global settings. To override the default setting, you must know the SCIM URL and the proxy type (Internet or On Premise).

- If the basic authentication is to be delegated to an on-premise user store, you must:
  - Set up the on-premise user store. See Using an On-Premise User Store for details on the different ways to set up the on-premise user store: https://help.hana.ondemand.com/help/frameset.htm?04cbd0f30d524612aa438ed0b0eed217.html.
  - Update the SCIM configuration to use "Cloud Platform SCIM".
- If the basic authentication is to be performed against a URL that supports HTTP basic authentication, you must:
  - Update the SCIM configuration to use "Mobile Service SCIM".
  - Specify that URL in the account security settings under Default Authentication Destination.

Note

Additionally, you can override the configured URL in the application configuration if you desire to delegate the basic authentication for a particular application to a different URL (but only if the global SCIM configuration in account security is configured to use "Mobile Service SCIM").

When you override the default, the web service verifies whether the SCIM URL you specify is accessible and that the back-end server is running.

Procedure

2. For the SCIM Type, select either:
   - Cloud Platform SCIM – delegates all basic authentication requests for the SAP Cloud Platform tenant account to the SCIM server that you define with the URL and Proxy Type. If you are using SAP cloud connector for authentication, set the URL to http://scc.scim:80/scim/v1 and set the proxy type to OnPremise.
   - Mobile Service SCIM – delegates basic authentication requests for the mobile service for development and operations applications to a URL that supports HTTP basic authentication. You can customize the URL at the application level.
3. Under **Default Authentication Destination**, enter:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
</table>
| URL         | The SCIM destination URL. Based on the value of **SCIM Type**, enter:  
  ○ Mobile Service SCIM – a URL that supports HTTP basic authentication.  
  See Using an On-Premise User Store. |
| Proxy Type  | Select either:  
  ○ **Internet** – destinations use an Internet proxy to connect to systems in the public domain that are accessible to everyone, for example, `http://www.google.com` or  
  ○ **OnPremise** – destinations connect to systems that are behind a firewall. Select if the configured URL is not accessible on the Internet, and only accessible via SAP cloud connector. |

4. Click **Save**.

**Related Information**

- User Identity Propagation Methods [page 183]
- Application Authentication [page 189]
- User Identity Propagation Methods [page 183]

**1.4.4.3 Managing Inbound Request Traffic**

Set thresholds for inbound registration requests, online requests, and offline requests. When the threshold is reached, requests are throttled and the server generates HTTP error code "429 Too Many Requests".

**Context**

Set the registration, online, and offline request thresholds at the application level, or the online request threshold at the connection level. You can view these settings for each application and connection. Monitor the inbound request traffic via the **Server Data Report**, and use the data to adjust thresholds to better control request traffic during busy periods.
Procedure

1. When you define an application, you can set threshold values in seconds for incoming registration requests, online requests, and offline requests. When you define a back-end connection, you can set threshold values for online requests. Leave blank (default) or set to 0 to remove a threshold. Set a threshold value from 1 - 2147483647.

Guidelines for managing inbound requests:
- For applications with low traffic requests, accept the default setting of 0.
- For applications with high traffic requests, consider setting different registration, online, and offline threshold settings.

2. Use the Server Data Report to view activity for one or more applications during various time periods to determine request traffic patterns. Use the information to identify periods of high request traffic that may burden the system.

Keep in mind:
- If you select multiple applications, the value is a sum value.
- If you select a single application or connection, the value is for only the selected application and connection.
- In most cases, the threshold hit count charts should have no data. This means the actual request count is lower than the threshold value, so the server is in a good status.
- If the charts show a lot of data, then it means the server load is too high or the threshold value is too small.
- If you leave the threshold blank or set it to 0, there is no limitation, so the threshold hit count chart will never have any data.

3. Adjust the request thresholds as needed to control request traffic.

1.4.4.4 Configuring a Secure Login Server Profile

Configure a secure login server profile to enable applications to use a certificate to access trusted services.

Prerequisites

You need the name and URL of the secure login server, and its profile name. You also need to set up a destination to the secure login server via the Destinations option. The destination must use the No Authentication method.

Context

Applications that are secured by SAML 2.0 authentication can retrieve a client certificate from the secure login server, and use it to access its trusted services. You can edit an existing secure login server profile, or you can create a new one.
1. In Mobile Service for Development and Operations cockpit, select [Settings > Secure Login Server].
   You can filter the existing profiles to find a specific one, or sort the profiles by column.

2. (Optional) Click Create Secure Login Server Profile, configure the profile, and click Save.

   ○ Profile Name – use the same profile name that is defined for the secure login server.
   ○ Destination Name – select or add the destination to the secure login server. You can only choose a
     destination that uses the No Authentication method.
   ○ Server URL – provide the internal secure login server URL.

3. (Optional) To edit an existing profile, select the settings icon under Actions, and select Edit.

4. (Optional) To delete an existing profile, select the settings icon under Actions, select Delete, and
   confirm.

1.4.5 Security Administration

An essential element when planning your mobile service for development and operations landscape is to
provide secure propagation of mobile users’ identities to back-end systems. The product supports a range of
popular application authentication protocols and maps them to back-end systems.

Several factors can help you determine the best security configuration for your mobile applications. The most
important factor is your current or planned landscape architecture. The security landscape includes
application authentication, transport and session security, and data protection and privacy. The high-level
landscape options are:

- Cloud – all components run on a cloud platform, either SAP Cloud Platform alone or with another cloud
  platform.

- Hybrid – components run on SAP Cloud Platform, and services run in an on-premise landscape.
mobile service for development and operations support mobile applications, and each application includes these security properties, which you can configure in Mobile Service for Development and Operations cockpit:

- A user authentication policy—also called a security configuration—defines the HTTPS protocol to authenticate the user, for example, basic authentication, SAML 2.0, OAuth 2.0, or X.509 user certificates. For more information about supported security configurations, see Application Authentication.
- One or more back-end HTTP endpoints—also called app URLs—define how an authenticated mobile user’s identity is relayed to back-end systems.

For simple authentication that uses HTTPS basic authentication headers, mobile services support the System for Cross-Domain Identity Management (SCIM) protocol.

SAP Gateway also works with the SCIM server, which is often the simplest, most secure authentication method to use for your hybrid cloud SAP environment.

User Identity Propagation Methods [page 183]

Design your system to provide secure propagation of mobile user identities to back-end systems. Select an identity propagation method based on your existing authentication model.

Application Authentication [page 189]

Application authentication verifies and validates the identities of application users.

Data Protection and Privacy [page 199]

SAP Cloud Platform mobile service for development and operations does not track or store personal data; it tracks data that is related to the mobile service and set-up details.

Transport and Session Security [page 200]

Mobile services and applications that connect using secure HTTP ensure that communication channels use encrypted connections.

End-to-End Security Configuration [page 201]

Configure secure communication between mobile applications and back-end systems in an SAP Cloud Platform mobile service for development and operations landscape.

1.4.5.1 User Identity Propagation Methods

Design your system to provide secure propagation of mobile user identities to back-end systems. Select an identity propagation method based on your existing authentication model.

The security configuration options that are available in SAP Cloud Platform mobile service for development and operations include basic authentication, OAuth, Form-based SAML, and X.509 user certificates. In addition, you can implement a single sign-on mechanism for each application.

Security Configuration for Authentication

One consideration is to select a security configuration option for authenticating users.

Basic Authentication

Basic authentication validates user credentials, typically, a user name and password.
You can delegate basic authentication requests to either the default identity provider or the SCIM server. The default identity provider is the SAP ID service in the cloud. You must have an account specifically for this service. To use the SCIM server, a tenant account can connect to SAP Cloud Platform mobile service for development and operations with their on-premise identity provider (typically AD/LDAP); this way, existing corporate users can access SAP Cloud Platform mobile service for development and operations apps using their existing accounts.

See:
- Authenticating with the Default Identity Provider
- Configuring SCIM Authentication

Form-Based SAML Authentication

To use the Security Assertion Markup Language 2.0 (SAML2) protocol in SAP Cloud Platform mobile service for development and operations, define a local service provider to determine authorization based on information from a trusted identity provider that maintains its own database of users and their SAML2 attributes. When a user requests access to a resource, the service provider retrieves a SAML2 assertion from the identity provider and reads the user name from either the subject or one of the SAML2 assertion attributes to determine whether to grant the user access to the resource.

To use SAML-based authentication, select SAML as the security configuration type when you define an application.

Common identity providers used for SAML authentication are Microsoft Active Directory Federation Services versions 2.0 and 3.0.

OAuth Authentication

OAuth enables users to grant third-party access to their resources using token credentials, so clients need not share their passwords with resource owners. SAP Cloud Platform mobile service for development and operations supports OAuth for hybrid applications and for native applications built using SAP Cloud Platform SDK for iOS.

See Managing OAuth Authentication.

X.509 User Certificates

To authenticate X.509 user certificates, install appropriate trusted CA certificates in the SAP Cloud Platform infrastructure.

See Configuring X.509 Certificate Authentication.

Single Sign-on Mechanisms

Another consideration is to implement a single sign-on mechanism for user propagation. An application can be assigned one SSO mechanism at a time.

<table>
<thead>
<tr>
<th>SSO Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application-to-App</td>
<td>Enables mobile services to propagate user identities to other applications, which are consumed (deployed or subscribed) in the same SAP Cloud Platform account. A user identity is propagated to the application that is specified in the URL.</td>
</tr>
</tbody>
</table>

SAP Cloud Platform Mobile Service for Development and Operations

SAP Cloud Platform Mobile Service for Development and Operations
### SSO Mechanism

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Tip</strong></td>
</tr>
<tr>
<td>To find the URL, open SAP Cloud Platform Cockpit, and on the Overview page, see Application URL.</td>
</tr>
</tbody>
</table>

**Requirements:**
- The proxy type for the destination must be Internet.
- The application that is receiving the SSO connection (receiving application) must be consumable from the same SAP Cloud Platform account – either through deployment or a subscription.
- The receiving application must be either a Java or an HTML5 application.
- When developing a Java application, see Enabling Authentication for information about enabling the application to accept application-to-application SSO.

Configure your account to allow principal propagation; see the “Specifying Custom Local Provider Settings” section in ID Federation with the Corporate Identity Provider.

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

- To configure ApptoAppSSO for an application not hosted on the same SAP Cloud Platform account; see the saml2_audience section in Application-to-Application SSO Authentication.

### SAPAssertionSSO

Configure the back-end system to accept SAP assertion tickets that are signed by a trusted X.509 DSA key pair, and define these properties.

**Note**
You can either select a PKCS #12 file, in which case, the certificate and signing key properties are populated automatically, or you can define these properties manually. The certificate must be a Digital Signature Algorithm (DSA).

- **PKCS #12 File** – click Browse and select a PKCS #12 file that contains the certificate and signing key.
- **File Password** – password to access the PKCS #12 file.
- **Issuer SID** – system ID of the certificate issuer; must be trusted by the back-end system.
- **Issuer Client** – client ID of the certificate issuer; must be trusted by the back-end system. For a HANA XS back end, the same value as the Issuer SID.
- **Recipient SID** – back-end system ID. For a HANA XS back end, you can find the SID at the top of the Admin console. In an SAP Gateway system, you can find the value in the connection properties, for example, SAPgui.
- **Recipient Client** – client ID of the back-end system. For a HANA XS back end, enter 000.
- **Certificate** and **Signing Key** – generate the values using openssl:
### SSO Mechanism | Description
---|---
1. **Run:**
   - `openssl dsaparam -out dsaparam.pem 1024`
   - `openssl gendsa -out dsaprivkey.pem dsaparam.pem`
   - `openssl req -new -x509 -key dsaprivkey.pem -out dsacert.pem`
   - `openssl pkcs8 -topk8 -inform PEM -outform PEM -in dsaprivkey.pem -out dsaprivkeyonly.pem -nocrypt`
2. **Principal Propagation** *(If proxy type is OnPremise)*
   - Allows destinations to forward the identity of an on-demand user to SAP cloud connector, which then forwards it to the back-end system of the relevant on-premise system. An on-demand user need not provide his or her identity for each connection to an on-premise system when using SAP Cloud Connector.
   - You can use the principal propagation authentication type in hybrid cloud configurations that use SAP cloud connector, which can dynamically transform a user’s identity to an X.509 user certificate. SAP Gateway Java and SAP Gateway ABAP servers typically host the services that support principal propagation via SAP Cloud Connector. Configuration is required in SAP Cloud Connector as well as the SAP Gateway server to establish trust between the two for principal propagation to happen.
   - To learn more about configuring this trust relationship, see [SAP Cloud Platform Cloud Connector](https://help.hana.ondemand.com/help/frameset.htm?e6c7616abb5710148cfcf3e75d96d596.html)
3. **OAuth2SAMLBearerAssertion**
   - Enables applications to use SAML assertions to access OAuth-protected resources.
   - **Enter:**
     - **Audience** – intended assertion audience, which is verified by the target OAuth authorization server.
     - **Client Key** – key that identifies the consumer to the authorization server.
     - **Token Service URL** – URL of the OAuth server.
     - **Token Service User** – user for basic authentication to the OAuth server.
     - **Token Service Password** – password for the token service user.
     - **System User** – user who requests an access token from the OAuth authorization server. If this property is not specified, the currently logged-in user is used.
     - **Security Domain Qualifier** – security domain of the user for whom the access token is requested.
     - **Company Identifier** – company identifier.
<table>
<thead>
<tr>
<th>SSO Mechanism</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✅ SAML Assertion Issuer</td>
<td>✗ SAML Assertion Issuer – issuer of the SAML assertion.</td>
</tr>
<tr>
<td>✗ Authentication Class</td>
<td>✗ Authentication Class – value of the AuthnContextClassRef tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication. See the SAML 2.0 specification.</td>
</tr>
<tr>
<td>✗ Name ID Format</td>
<td>✗ Name ID Format – value of the NameIdFormat tag, which is part of the generated OAuth2 SAML Bearer Assertion authentication.</td>
</tr>
<tr>
<td>✗ Define User as Name ID</td>
<td>✗ Define User as Name ID – if the value of the NameId tag is undefined, the generated SAML2 assertion uses the currently logged-in user.</td>
</tr>
</tbody>
</table>

| Basic Authentication                | Enter a user name and password to access the back-end system. If you do not provide a user name and password, and the mobile service for development and operations authenticates the end-user credentials using Basic, the user name and password credentials are propagated to the back end. |

<table>
<thead>
<tr>
<th>Client Certification Authentication</th>
<th>Requires a client certificate that is signed by a certificate authority (CA) trusted by back-end systems. It uses a technical user certificate to perform mutual SSL authentication. The requirements include:</th>
</tr>
</thead>
<tbody>
<tr>
<td>✗ Proxy type must be Internet.</td>
<td>✗ Proxy type must be Internet.</td>
</tr>
<tr>
<td>✗ Back-end URL must use HTTPS.</td>
<td>✗ Back-end URL must use HTTPS.</td>
</tr>
<tr>
<td>✗ You must provide values for both the keystore and truststore parameters:</td>
<td>✗ You must provide values for both the keystore and truststore parameters:</td>
</tr>
<tr>
<td>✗ Keystore – contains the client certificate that should be used for mutual SSL authentication.</td>
<td>✗ Keystore – contains the client certificate that should be used for mutual SSL authentication.</td>
</tr>
<tr>
<td>✗ Truststore – contains the certificate authority that issued the server certificate for the back-end system, as a trusted certificate authority entry.</td>
<td>✗ Truststore – contains the certificate authority that issued the server certificate for the back-end system, as a trusted certificate authority entry.</td>
</tr>
</tbody>
</table>

**Note**

The configured technical user certificate is always used to establish a mutual SSL connection. Even if the user is authenticated to the SAP Cloud Platform mobile service for development and operations using the certificate, it will not be forwarded to the back end.

| No Authentication                  | Back ends require no credentials for authentication. Your destination is granted direct access to the relevant on-premise service.                                                                 |

---

**Security Configuration and SSO Mechanisms**

Make sure the single sign-on mechanism is appropriate for the security configuration option selected for the application.

*Security Configuration Options and SSO Mechanisms* provides a matrix of security configuration options and the single sign-on mechanisms available to implement with them. The columns represent the security configuration options available for applications. The rows represent the SSO mechanisms. An “X” indicates the propagation methods that can be implemented for each security type. Technical User indicates the user name and password that is configured for a single sign-on mechanism, and enables SAP Cloud Platform mobile
service for development and operations to access the back end. A Real User is typically an individual accessing from a device with a user name and password.

Table 65: Security Configuration Options and SSO Mechanisms

<table>
<thead>
<tr>
<th>Security Configuration Options (columns)</th>
<th>SSO Mechanisms (rows)</th>
<th>None</th>
<th>Basic</th>
<th>Form (SAML)</th>
<th>Certificate</th>
<th>OAuth</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Authentication</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Basic Auth (Technical User)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Basic Auth (Real User)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Certificate (Technical User)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SAPAssertionSSO</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>App-to-AppSSO</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>OAuth2SAML-BearerAssertion</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Principal Propagation (valid only when using Cloud-Connector)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Parent topic: Security Administration [page 182]

Related Information

Application Authentication [page 189]
Data Protection and Privacy [page 199]
Transport and Session Security [page 200]
End-to-End Security Configuration [page 201]
Configuring SCIM Authentication [page 179]
Application Authentication [page 189]
Defining Application Security [page 92]
Configuring SCIM Authentication [page 179]
Setting Up One-Time Password Validation [page 198]
1.4.5.2 Application Authentication

Application authentication verifies and validates the identities of application users.

SAP Cloud Platform mobile service for development and operations use the same identity provider (IdP) configuration that SAP Cloud Platform uses for SAML authentication. For information about how to change the default identity provider configuration, navigate to ID Federation with the Corporate Identity Provider and read Using an IdP Different from the Default.

After you configure the identity provider, select one of the following single sign-on authentication mechanisms for your application to access back-end systems. An application can be assigned only one SSO Mechanism at a time. To replace an existing SSO Mechanism, you must delete it, and then create a new one.

Table 66: SSO Authentication Support Matrix

<table>
<thead>
<tr>
<th>Security Configuration</th>
<th>(SSO) Authentication Mechanism to the Back End</th>
</tr>
</thead>
</table>
| None. No authentication challenges are performed by the application. | ● No authentication (anonymous access, no challenge)  
● Technical user name and password (basic challenge)  
● Technical user certificate (X.509 challenge) |
| Basic. User name and password.  
Default Basic Authentication delegates Basic authentication to the SAP ID service. If you delegate to a different on-premise user store or to an HTTP URL that supports Basic authentication, verify the SCIM settings. | ● No authentication (anonymous access, no challenge)  
● Technical user name and password (basic challenge)  
● Real user name and password (basic challenge)  
● Technical user certificate (X.509 challenge)  
● SAPAssertionSSO  
● App-to-AppSSO  
● OAuth2SAMLBearerAssertion  
● User identity (Principal Propagation) |

Note

If you configure a custom IdP or Cloud Identity tenant account in SAP Cloud Platform Cockpit, only SAML authentication is delegated to that provider and has no effect on Basic authentication.

If you configure Basic for a real end user, you must leave the technical user name and password fields blank.
## Security Configuration

<table>
<thead>
<tr>
<th>SAML</th>
<th>Uses SAML 2.0, which SAP Cloud Platform provides, and the SAP ID service to authenticate users against both SAP user accounts and SCN accounts. Subscribers can use SAP Cloud Platform Cockpit to customize identity provider configurations for their SAML 2.0 provider. SAML 2.0 configuration is global at the account/subscription level. All applications that are configured for SAML authentication use the same IdP. Certificate. Enables authenticating users with client certificates. By default, mobile services do not request client certificates during authentication. The URL for certificate authentication should be .cert.&lt;landscape_identifier&gt;.hana.ondemand.com, where &lt;landscape_identifier&gt; is similar to us1, eu1, and so forth. OAuth. An open protocol for secure authorization of applications using a standard method. SAP Cloud Platform must be the authorization server. <strong>Note</strong> SAP Cloud Platform mobile service for development and operations supports OAuth for hybrid applications and for native applications built using SAP Cloud Platform SDK for iOS.</th>
<th>(SSO) Authentication Mechanism to the Back End</th>
</tr>
</thead>
<tbody>
<tr>
<td>• No authentication (anonymous access, no challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user name and password (basic challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user certificate (X.509 challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SAPAssertionSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• App-to-AppSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• OAuth2SAMLBearerAssertion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• User identity (Principal Propagation)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• No authentication (anonymous access, no challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user name and password (basic challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user certificate (X.509 challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SAPAssertionSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• App-to-AppSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• OAuth2SAMLBearerAssertion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• User identity (Principal Propagation)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• No authentication (anonymous access, no challenge, principal propagation)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user name and password (basic challenge, principal propagation)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• Technical user certificate (X.509 challenge)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• SAPAssertionSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• App-to-AppSSO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• OAuth2SAMLBearerAssertion</td>
<td></td>
<td></td>
</tr>
<tr>
<td>• User identity (Principal Propagation)</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

---

### Configuring Anonymous Access [page 191]
Configure applications so that no authentication challenges are sent, and all application requests are processed anonymously.

### Authenticating With the Default Identity Provider [page 192]
For applications that use basic authentication, you can configure SAP Cloud Platform mobile service for development and operations to authenticate users with the default identity provider.

### Configuring OAuth Authentication [page 192]
OAuth is an authentication protocol. Using SAP Cloud Platform as the authorization server, the OAuth framework enables hybrid applications to obtain limited access to HTTP services.

### Managing OAuth Authentication [page 193]
Using SAP Cloud Platform as the authorization server, the OAuth protocol enables hybrid applications to obtain limited access to HTTP services. SAP Cloud Platform mobile service for development and operations supports OAuth for hybrid applications and for native applications built using SAP Cloud Platform SDK for iOS.

### Publishing an OAuth Application [page 196]
You can publish an OAuth application by using custom properties or writing custom JSON code.
Configuring X.509 Certificate Authentication [page 197]
Enable SAP Cloud Platform mobile service for development and operations to authenticate clients by challenging and validating X.509 client certificates.

Setting Up One-Time Password Validation [page 198]
A one-time password (OTP) strategy increases the security of applications by requiring the user to know two secrets instead of just one. Setting up OTP requires both the tenant administrator and the SAP Cloud Platform administrator.

Parent topic: Security Administration [page 182]

Related Information
- User Identity Propagation Methods [page 183]
- Data Protection and Privacy [page 199]
- Transport and Session Security [page 200]
- End-to-End Security Configuration [page 201]
- Defining Application Security [page 92]
- Configuring SCIM Authentication [page 179]
- Setting Up One-Time Password Validation [page 198]
- User Identity Propagation Methods [page 183]

1.4.5.2.1 Configuring Anonymous Access

Configure applications so that no authentication challenges are sent, and all application requests are processed anonymously.

Context
Reserve anonymous access for applications that do not access sensitive corporate systems.

Procedure
1. In Mobile Service for Development and Operations cockpit, select Applications, and click the Create Application icon.
2. For Security Configuration, choose None, and click Save.
1.4.5.2.2 Authenticating With the Default Identity Provider

For applications that use basic authentication, you can configure SAP Cloud Platform mobile service for development and operations to authenticate users with the default identity provider.

Prerequisites

An account with the SAP ID service in the cloud.

Procedure

2. For SCIM Type, select Default Identity Provider.
3. Click Save.

1.4.5.2.3 Configuring OAuth Authentication

OAuth is an authentication protocol. Using SAP Cloud Platform as the authorization server, the OAuth framework enables hybrid applications to obtain limited access to HTTP services.

Context

OAuth service providers can grant to third-parties access to their resources without sharing their passwords.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications, and click the Create Application icon.
2. For Security Configuration, select OAuth.
3. Under OAuth Settings, enter:
Table 67:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client ID</td>
<td>An automatically generated random ID that identifies the application client to the authorization server. To regenerate the Client ID, select Regenerate ID.</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the access token is valid.</td>
</tr>
<tr>
<td>Refresh Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the refresh token is valid.</td>
</tr>
<tr>
<td>Authorization Endpoint</td>
<td>(Read only) URL that is automatically retrieved from the server; the service authenticates the user and provides an authorization code.</td>
</tr>
<tr>
<td>Token Endpoint</td>
<td>(Read only) URL that is automatically retrieved from the server; the service exchanges the authorization code, obtained from the authorization endpoint, for an access token.</td>
</tr>
<tr>
<td>End-User UI</td>
<td>(Read only) URL that is automatically retrieved from the server; the service manages the issued access token.</td>
</tr>
</tbody>
</table>

4. Click Save.

Related Information

Managing OAuth Authentication [page 193]

1.4.5.2.4 Managing OAuth Authentication

Using SAP Cloud Platform as the authorization server, the OAuth protocol enables hybrid applications to obtain limited access to HTTP services. SAP Cloud Platform mobile service for development and operations supports OAuth for hybrid applications and for native applications built using SAP Cloud Platform SDK for iOS.

Use SAP Cloud Platform mobile service for development and operations to create, modify, or delete OAuth authentication settings. SAP Cloud Platform mobile service for development and operations also lets you use multiple OAuth client IDs. OAuth service providers can grant access to resources to third parties without sharing their passwords.

Related Information

Creating OAuth Authentication Setting [page 194]
Editing OAuth Authentication Setting [page 195]
Deleting an OAuth Authentication Setting [page 195]
1.4.5.2.4.1 Creating OAuth Authentication Setting

In SAP Cloud Platform mobile service for development and operations you can create OAuth authentication settings. You can also add multiple OAuth client IDs.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Applications.
2. Select an existing application, and click Actions Configure.
3. Under OAuth Settings, click the Add icon.
4. In Add OAuth Settings, enter:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Client ID</td>
<td>An automatically generated random ID that identifies the application client to the authorization server. To regenerate the Client ID, select Regenerate ID.</td>
</tr>
<tr>
<td>Redirect URL</td>
<td>Enter a valid Redirect URL.</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the access token is valid.</td>
</tr>
<tr>
<td>Refresh Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the refresh token is valid.</td>
</tr>
<tr>
<td>Authorization Endpoint</td>
<td>(Read only) URL that is automatically retrieved from the server; the service authenticates the user and provides an authorization code.</td>
</tr>
<tr>
<td>Token Endpoint</td>
<td>(Read only) URL that is automatically retrieved from the server; the service exchanges the authorization code, obtained from the authorization endpoint, for an access token.</td>
</tr>
<tr>
<td>End-User UI</td>
<td>(Read only) URL that is automatically retrieved from the server; the service manages the issued access token.</td>
</tr>
</tbody>
</table>

5. Click Save.

The OAuth Authentication settings get created under the OAuth Settings table.

Related Information

Managing OAuth Authentication [page 193]
1.4.5.2.4.2 Editing OAuth Authentication Setting

In SAP Cloud Platform mobile service for development and operations, you can modify existing OAuth authentication settings.

Procedure

1. Under OAuth Settings, click the Edit icon.
2. In Edit OAuth Settings, you can modify only the following:

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redirect URL</td>
<td>Enter a valid Redirect URL.</td>
</tr>
<tr>
<td>Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the access token is valid.</td>
</tr>
<tr>
<td>Refresh Token Lifetime</td>
<td>Specify the number of days, hours, or minutes for which the refresh token is valid.</td>
</tr>
</tbody>
</table>

3. Click Save.

Related Information

Managing OAuth Authentication [page 193]

1.4.5.2.4.3 Deleting an OAuth Authentication Setting

In SAP Cloud Platform mobile service for development and operations, you can delete an existing OAuth authentication setting.

Procedure

1. Under OAuth Settings, select the OAuth settings you want to delete.
2. Click the Delete icon.
3. In the Confirm Delete pop-up, select OK.
Related Information

Managing OAuth Authentication [page 193]

1.4.5.2.5 Publishing an OAuth Application

You can publish an OAuth application by using custom properties or writing custom JSON code.

Context

SAP Cloud Platform mobile service for development and operations allows you to publish an OAuth application in either of the following ways:

- **Custom Properties**
  Add custom properties and corresponding values to publish the OAuth application.

  Table 70: Sample Custom Properties

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>oauth2.clientID</td>
<td>&quot;6730a9b7-aeff-4c7a-8b92-0ef3eac7d4f8&quot;</td>
</tr>
<tr>
<td>oauth2.authorizationEndpoint</td>
<td>&quot;<a href="https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/authorize">https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/authorize</a>&quot;</td>
</tr>
<tr>
<td>oauth2.tokenEndpoint</td>
<td>&quot;<a href="https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/token">https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/token</a>&quot;</td>
</tr>
<tr>
<td>oauth2.endUserUI</td>
<td>&quot;<a href="https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2">https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2</a>&quot;</td>
</tr>
<tr>
<td>oauth2.redirectUrl</td>
<td>&quot;<a href="http://www.mycallback.com">http://www.mycallback.com</a>&quot;</td>
</tr>
<tr>
<td>oauth2.grantType</td>
<td>authorization_code</td>
</tr>
</tbody>
</table>

- **Custom JSON**
  Add your json code to publish the OAuth application.

  ```json
  {"oauth2.clientID":"6730a9b7-aeff-4c7a-8b92-0ef3eac7d4f8","oauth2.authorizationEndpoint":"https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/authorize","oauth2.tokenEndpoint":"https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2/api/v1/token","oauth2.endUserUI":"https://oauthasservices-x5cd579a4.neo.ondemand.com/oauth2","oauth2.redirectUrl":"http://www.mycallback.com","oauth2.grantType":"authorization_code"}
  ```
Procedure

1. In Mobile Service for Development and Operations cockpit, select *Applications*.
2. Select an existing application, and click *Actions > Publish*.
3. To publish the OAuth application using custom properties, select *Custom Properties*.
   a. Select the Create icon.
   b. Enter a *Key* and *Value*.
   c. Click *Save*.
4. To publish the OAuth application using JSON code, select *Custom JSON*.
   a. Enter the JSON code for publishing your application.
   b. Click *Save*.

1.4.5.2.6 Configuring X.509 Certificate Authentication

Enable SAP Cloud Platform mobile service for development and operations to authenticate clients by challenging and validating X.509 client certificates.

Prerequisites

To enable client certificate authentication, install appropriate trusted CA certificates in the SAP Cloud Platform infrastructure.

Procedure

1. Connect your mobile device to the `<scpms-subscription>.cert.<landscape_identifier>.hana.ondemand.com` domain for client certificate authentication, where `<landscape_identifier>` is similar to us1, eu1, and so forth. See *Enabling Client Certificate Authentication*.
   
   By default, the SAP Cloud Platform load balancer trusts all the CAs listed in *Trusted Certificate Authorities for Client Certificate Authentication*.

2. Create a Java keystore called `hcpms_trusted_ca.jks` for your trusted CA certificate by running on a command line:

   ```bash
   keytool -import -trustcacerts -alias sapsso -file SSO_CA.cer -keystore hcpms_trusted_ca.jks
   ```

3. Upload the keystore in the subscriber account, for example, run:
   
   ○ Production account:

   ```bash
   neo.sh upload-keystore --account <ConsumerAccountName>
   ```
1.4.5.2.7 Setting Up One-Time Password Validation

A one-time password (OTP) strategy increases the security of applications by requiring the user to know two secrets instead of just one. Setting up OTP requires both the tenant administrator and the SAP Cloud Platform administrator.

Prerequisites

- The tenant administrator must have access to the SAP Cloud Platform Identity Authentication service tenant console. In this procedure, this role is referred to as the SAP Cloud Platform Identity Authentication administrator.
- You must establish SAML trust between the SAP Cloud Platform account and SAP Cloud Platform Identity Authentication service tenant. In summary:
  - On the SAP Cloud Platform side, configure trust to the SAP Cloud Platform Identity Authentication service, as described in ID Federation with the Corporate Identity Provider.
  - On the SAP Cloud Platform Identity Authentication service side:
    - Download the tenant SAML 2.0 metadata using information in Tenant SAML 2.0 Configuration.
    - Configure trust to a service provider using information in Configure a Trusted Service Provider.
- Mobile app users must have an OTP generator—such as SAP Authenticator, Google Authenticator, or Microsoft Authenticator—installed on their devices. Typically, the user activates two-factor authentication from the mobile device user profile page. Once activated, the passcodes used in two-factor authentication are generated. For more information, see Two-Factor Authentication.

Context

With two-factor authentication, the user is required to provide a one-time password, also called a passcode, in addition to primary credentials. Passcodes are time-based and valid for one login attempt, thus providing additional security to the static passwords. Passcodes are generated by an authenticator application that is installed on the mobile device. For two-factor authentication, you must set up a one-time password validation service.
Procedure

1. (SAP Cloud Platform Identity Authentication administrator) Create a technical user using information in Add Administrators. Give the technical user a name, such as RestClient, and set the password.

Communicate the technical user name and password to the SAP Cloud Platform administrator.

2. (SAP Cloud Platform Identity Authentication administrator) Configure the risk-based authentication rules. When a user tries to access the application, these rules evaluate whether the user meets authentication criteria. For more information, see Configure Risk-Based Authentication.

3. (SAP Cloud Platform administrator) Create a destination for the one-time password validation service.

   ○ Name – enter otpValidationService.
   ○ Type – enter HTTP.
   ○ URL – enter the destination URL, in the format: https://<scptenant>.<accountsxxx>.ondemand.com/service/users/otp. Use Check Connection to verify the link connection.
   ○ Proxy Type – enter Internet.
   ○ Authentication – enter BasicAuthentication.
   ○ User – enter the technical user name that you created in step 1, such as RestClient.
   ○ Password – enter the password for the technical user.
   ○ Use default JDK truststore – indicate whether to use the default truststore.

Next Steps

In Mobile Service for Development and Operations cockpit, configure the application for one-time password validation:

1. Select an application, and then select Security under Assigned Features.
2. Under Applications Settings, select Require One Time Passcode.

Related Information

Application Authentication [page 189]
User Identity Propagation Methods [page 183]

1.4.5.3 Data Protection and Privacy

SAP Cloud Platform mobile service for development and operations does not track or store personal data; it tracks data that is related to the mobile service and set-up details.

Mobile service and set-up details typically include:
• Tenant-specific platform configuration data – stored in a tenant proprietary HANA database.
• Application client logs and usage reports – maintained by SAP Cloud Platform mobile service for development and operations; no sensitive data is stored. If a developer programmatically alters the data model, any person-related data that is tracked or used must comply with the data protection rules of the target countries. This includes authentication, authorization, and encryption details. Tenants are required to both secure this data and log all access to it.
• Applications that you create – may store sensitive data in an offline data store. You must set the client password policy that is used to unlock the data store when an application initializes. See Defining Client Password Policies. Developers should also encrypt the contents of the offline data store using the storeEncryptionKey method. When the store is used for the first time, it is automatically encrypted.

Related Information

User Identity Propagation Methods [page 183]
Application Authentication [page 189]
Transport and Session Security [page 200]
End-to-End Security Configuration [page 201]
Defining Client Password Policy [page 52]

1.4.5.4 Transport and Session Security

Mobile services and applications that connect using secure HTTP ensure that communication channels use encrypted connections.

Follow these recommendations:

• All destination URLs should use SSL encryption with the HTTPS protocol, even when the applications access on-premise services through the SAP cloud connector.
• Test SSL connections at an early stage of application development; implementing HTTPS can cause some issues that should be identified and addressed as early as possible.
• For Internet destinations, enable mutual authentication for technical users, then configure a certificate alias in the SAP Cloud Platform mobile service for development and operations back-end system.
• For increased session security, enable client session cookies on the platform. After a user is authenticated, subsequent requests contain cookies that can be executed without requiring the user to log in again, as long as the session is still valid. Session validity makes session cookies less prone to malicious use.

Parent topic: Security Administration [page 182]
1.4.5.5   **End-to-End Security Configuration**

Configure secure communication between mobile applications and back-end systems in an SAP Cloud Platform mobile service for development and operations landscape.

**Configure Secure Access to SAP NetWeaver** [page 201]
Configure secure communication between SAP Cloud Platform mobile service for development and operations mobile applications and SAP NetWeaver back-end systems, using X.509 certificates and single sign-on.

**Related Information**

- User Identity Propagation Methods [page 183]
- Application Authentication [page 189]
- Data Protection and Privacy [page 199]
- End-to-End Security Configuration [page 201]
- Defining Connectivity [page 58]
- Defining Connectivity [page 58]

**1.4.5.5.1   Configure Secure Access to SAP NetWeaver**

Configure secure communication between SAP Cloud Platform mobile service for development and operations mobile applications and SAP NetWeaver back-end systems, using X.509 certificates and single sign-on.

To configure security, you need:

- Access to SAP Service Marketplace. To request a user ID, navigate to [SAP Service Marketplace](https://service.sap.com).
- A tool to create certificates such as OpenSSL, or a certificate authority. You can download OpenSSL for Windows from [SourceForge.Net](http://sourceforge.net).
- Chrome Web browser.
- SAP cloud connector.
- Postman REST Client. In Chrome, navigate to the Chrome Web Store, search for "postman rest client", and download Postman.

Securing communication between SAP Cloud Platform mobile service for development and operations applications and SAP Gateway requires that you perform some configuration tasks on the server machine and some on the SAP NetWeaver back-end system.

<table>
<thead>
<tr>
<th>Server-Side Tasks</th>
<th>Back-End Tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create a self-signed root certificate authority</td>
<td>Enable SSO on an SAP NetWeaver back end</td>
</tr>
<tr>
<td>Create an intermediate certificate</td>
<td>Import the root CA into SAP NetWeaver</td>
</tr>
<tr>
<td>Import the intermediate certificate into SAP cloud connector</td>
<td>Map user certificates in SAP NetWeaver</td>
</tr>
<tr>
<td>Configure SAP cloud connector for principal propagation</td>
<td>Activate and test a SAP NetWeaver back-end service</td>
</tr>
<tr>
<td>Enable verbose logging</td>
<td>(Optional) Configure a Java back end</td>
</tr>
<tr>
<td>Create an application to use X.509 and SAP NetWeaver</td>
<td></td>
</tr>
<tr>
<td>Test the application connection to SAP NetWeaver</td>
<td></td>
</tr>
<tr>
<td>Review the SAP cloud connector logs</td>
<td></td>
</tr>
</tbody>
</table>

1. **Enabling SSO for an SAP NetWeaver System [page 203]**
   - Enable single sign-on (SSO) to your SAP NetWeaver system.

2. **Creating a Self-Signed Root Certificate Authority [page 204]**
   - Either create a self-signed certificate authority (CA), or use an existing one. If you use your own CA, you will need the CA certificate later.

3. **Creating Intermediate Certificates [page 204]**
   - Create an intermediate certificate to sign a short-lived user certificate on the SAP cloud connector after a user has been authenticated.

4. **Importing Certificates into SAP Cloud Connector [page 206]**
   - Import both the SAP cloud connector root CA and the intermediate CA into SAP cloud connector.

5. **Importing the Root CA into the Back-End System [page 207]**
   - Import the root CA that you created into the SAP NetWeaver back-end system.

6. **Configuring SAP Cloud Connector for Principal Propagation [page 207]**
   - Configure SAP cloud connector to use the Principal Propagation single sign-on provider to generate a short-lived certificate for access to SAP NetWeaver back-end systems.

7. **Enabling Verbose Logging [page 208]**
   - Enable verbose logging to make debugging easier.

8. **Mapping User Certificates in SAP NetWeaver [page 209]**
   - A certificate DN has a full set of attributes that you must map in SAP NetWeaver. Map both the end-user and the technical-user certificates.

9. **Activating and Testing SAP Gateway Services [page 211]**
    -Activate and test the SAP Gateway service that you want to run.

10. **Creating a Test Application [page 214]**
    -Create an application to test your security configuration.

11. **Testing Application Connections to SAP NetWeaver [page 215]**
    -Test that the application you created can connect to the SAP NetWeaver system.
12. **Reviewing SAP Cloud Connector Logs [page 216]**
   Review the logs and trace files that SAP cloud connector generates.

13. **Configuring Java Back-ends [page 217]**
   If the back-end system is an SAP Gateway Java instance, edit the instance profile, and set the configuration properties.

### 1.4.5.5.1.1 Enabling SSO for an SAP NetWeaver System

Enable single sign-on (SSO) to your SAP NetWeaver system.

#### Procedure

1. Open SAP Logon, and log in to the SAP NetWeaver system.
2. Run the RZ11 transaction, and for each parameter in the table below, enter the parameter name and value:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>icm/HTTPS/verify_client</td>
<td>1</td>
</tr>
<tr>
<td>icm/HTTPS/trust_client_with_subject</td>
<td>*</td>
</tr>
<tr>
<td>icm/HTTPS/trust_client_with_issuer</td>
<td>*</td>
</tr>
<tr>
<td>ssl/ssl_lib</td>
<td><code>&lt;Full path&gt;</code>\sapcrypto.dll</td>
</tr>
<tr>
<td>sec/libsapsecu</td>
<td><code>&lt;Full path&gt;</code>\sapcrypto.dll</td>
</tr>
<tr>
<td>ssf/ssfapi_lib</td>
<td><code>&lt;Full path&gt;</code>\sapcrypto.dll</td>
</tr>
<tr>
<td>ssf/name</td>
<td>SAPSECULIB</td>
</tr>
</tbody>
</table>

3. Click **Save**.

**Task overview:** Configure Secure Access to SAP NetWeaver [page 201]

**Next task:** Creating a Self-Signed Root Certificate Authority [page 204]
1.4.5.5.1.2 Creating a Self-Signed Root Certificate Authority

Either create a self-signed certificate authority (CA), or use an existing one. If you use your own CA, you will need the CA certificate later.

Prerequisites

On the Windows machine where SAP Cloud Platform mobile service for development and operations is installed, download OpenSSH or another tool that supports Linux commands. Run the commands using an account for which you have Administrator privileges.

Procedure

1. In a Linux command window, run:

```
openssl genrsa -des3 -out <path>\HCC_CA.key 1024
openssl req -sha256 -new -x509 -days 9999 -key <path>\HCC_CA.key -out <path>\HCC_CA.crt
openssl pkcs12 -export -clcerts -in <path>\HCC_CA.crt -inkey <path>\HCC_CA.key -out <path>\HCC_CA.p12
```

2. Enter the values requested by openssl.

Note the key length and verify that each system supports it.

Task overview: Configure Secure Access to SAP NetWeaver [page 201]

Previous task: Enabling SSO for an SAP NetWeaver System [page 203]

Next task: Creating Intermediate Certificates [page 204]

1.4.5.5.1.3 Creating Intermediate Certificates

Create an intermediate certificate to sign a short-lived user certificate on the SAP cloud connector after a user has been authenticated.

Context

The intermediate certificate must include a KEYCERTSIGN property.
Procedure

1. In a Linux command window, run:

   ```
   touch <path>/certindex
   echo 1000 > <path>/certserial
   echo 1000 > <path>/crlnumbe
   ```

2. Create a CA configuration file called `ca.conf`, and enter the following lines. Set the value of `dir` for your operating system and file structure:

   ```
   # vim ca.conf
   [ ca ]
   default_ca = myca
   [ crl_ext ]
   issuerAltName=issuer:copy
   authorityKeyIdentifier=keyid:always
   [ myca ]
   # Linux
dir = ./
   # Windows - change this value to the working path for this guide
   # dir = C:\OpenSSL-Win64\bin\<working directory>
   new_certs_dir = $dir
   unique_subject = no
   certificate = $dir/HCC_CA.crt
   database = $dir/certindex
   private_key = $dir/HCC_CA.key
   serial = $dir/certserial
   default_days = 730
   default_md = shal
   policy = myca_policy
   x509_extensions = myca_extensions
   crlnumber = $dir/crlnumber
   default_crl_days = 730
   [ myca_policy ]
   commonName = supplied
   stateOrProvinceName = supplied
   countryName = optional
   emailAddress = optional
   organizationName = supplied
   organizationalUnitName = optional
   [ myca_extensions ]
   basicConstraints = critical,CA:TRUE
   keyUsage = critical,any
   subjectKeyIdentifier = hash
   authorityKeyIdentifier = keyid:always,issuer
   keyUsage = digitalSignature,keyEncipherment,cRLSign,keyCertSign
   extendedKeyUsage = serverAuth
   [ v3_ca ]
   basicConstraints = critical,CA:TRUE,pathlen:0
   keyUsage = critical,any
   subjectKeyIdentifier = hash
   authorityKeyIdentifier = keyid:always,issuer
   keyUsage = digitalSignature,keyEncipherment,cRLSign,keyCertSign
   extendedKeyUsage = serverAuth
   ```

3. Create the intermediate key and the certificate signing request, and convert the client key to PKCS:

   ```
   openssl genrsa -out <path>/intermediate.key 1024
   openssl req -new -sha256 -key <path>/intermediate.key -out <path>/intermediate.csr
   openssl ca -batch -config <path>/ca.conf -notext -in <path>/intermediate.csr -out <path>/intermediate.crt
   openssl pkcs12 -export -clcerts -in <path>/intermediate.crt -inkey <path>/intermediate.key -out <path>/intermediate.p12
   ```
1.4.5.5.1.4 Importing Certificates into SAP Cloud Connector

Import both the SAP cloud connector root CA and the intermediate CA into SAP cloud connector.

**Prerequisites**

Verify that the KEYCERTSIGN property is defined in the intermediate CA.

**Context**

Import the certificates on the machine where SAP cloud connector is installed.

**Procedure**

1. In Internet Explorer, select *Internet Options ➤ Content ➤ Certificates*.
2. Select *Trusted Root Certification Authorities*, and import the `HCC_CA.crt` certificate.
1.4.5.5.1.5 Importing the Root CA into the Back-End System

Import the root CA that you created into the SAP NetWeaver back-end system.

Procedure

1. Log in to SAP GUI.
2. Start the `strust` transaction.
3. In the left column, select `SSL Server Standard`.
4. Click `Import Certificate`, select the root CA you created, and click `Add to Certificate List`.
5. Save your changes.

Task overview: Configure Secure Access to SAP NetWeaver [page 201]

Previous task: Importing Certificates into SAP Cloud Connector [page 206]

Next task: Configuring SAP Cloud Connector for Principal Propagation [page 207]

1.4.5.5.1.6 Configuring SAP Cloud Connector for Principal Propagation

Configure SAP cloud connector to use the Principal Propagation single sign-on provider to generate a short-lived certificate for access to SAP NetWeaver back-end systems.

Procedure

1. Log in to SAP cloud connector administration tool.
2. In the left column, select `Principal Propagation`, and click `Synchronize`.
3. In the top-right corner, click `Settings`.
4. In the left column, select `System Certificate`, then select the `P12 Certificate` option.
5. Click `Browse`, select the intermediate certificate file that you created, and import it.
6. Select `Principal Propagation`, and validate the certificate settings.

Note

After authentication, `<name>` is replaced with the user ID. You can add more attributes to the `Subject Pattern` string.
7. To create a sample end-user certificate for back-end user mapping:
   a. Click *Create Sample Certificate*.
   b. In the *CN name* field, enter any value, and click *Save*.

   The system generates the certificate.

**Task overview:** Configure Secure Access to SAP NetWeaver [page 201]

**Previous task:** Importing the Root CA into the Back-End System [page 207]

**Next task:** Enabling Verbose Logging [page 208]

### 1.4.5.5.1.7 Enabling Verbose Logging

Enable verbose logging to make debugging easier.

**Procedure**

1. Log in to SAP cloud connector administration tool.
2. In the left column, select *Audit*.
3. Under *Audit Settings*, select *All* as the *Audit Level*.
4. In the left column, select *Logs*.
5. Under *Log and Trace Settings*, select *All* as the *Log Level*.

**Task overview:** Configure Secure Access to SAP NetWeaver [page 201]

**Previous task:** Configuring SAP Cloud Connector for Principal Propagation [page 207]

**Next task:** Mapping User Certificates in SAP NetWeaver [page 209]
1.4.5.5.1.8 Mapping User Certificates in SAP NetWeaver

A certificate DN has a full set of attributes that you must map in SAP NetWeaver. Map both the end-user and the technical-user certificates.

Procedure

1. Open SAP Logon, and log in to the SAP NetWeaver system.
2. Run the SM30 transaction, and click Enter.
3. In the Maintain Table Views dialog, enter VUSREXTID as the Table/View.
4. Click Maintain, enter DN as the External ID type, and click the check mark.

5. In the certificate Details, find the value of Subject.

6. Click New Entries, and enter:
○ **External ID** – comma-separated list of the certificate Subject details.
○ **User** – ABAP ID.
○ **Activated** – select.

7. Repeat steps 5 and 6 to map certificates for both the end user and the technical user.

**Results**

The results should look similar to this:

```
    Assignment of External ID to Users
    +---------------------------------+-------+-----+
    | H. External ID                  | User  | Act.|
    | CN=ssodemo, OU=RIG, O=SAP, L=Greensboro, S=NC, C=US | ssodemo | ✓   |
    | CN=sstechnical, OU=RIG, O=SAP, L=Greensboro, S=NC, C=US | sstechnical | ✓   |
```

**Task overview:** Configure Secure Access to SAP NetWeaver [page 201]

**Previous task:** Enabling Verbose Logging [page 208]

**Next task:** Activating and Testing SAP Gateway Services [page 211]
1.4.5.1.9 Activating and Testing SAP Gateway Services

Activate and test the SAP Gateway service that you want to run.

Prerequisites

See the SAP Gateway topic Activate the Service.

Context

This example activates the `sap/opu/odata/iwfnd/RMTSAMPLEFLIGHT` service, which is provided with all Gateway systems.

Procedure

1. Open SAP Logon, and log in to the SAP Gateway system.
2. Run the SICF transaction.
3. For the Virtual Host, select default_host.
4. Under Maintain Services, click the Execute icon.
5. Click to the right of Service Path, and navigate to `sap/opu/odata/iwfnd/`. 
6. Double-click **RMTSAMPLEFLIGHT**, and select the **Logon Data** tab.
7. For the Procedure, select **Required with SSL Certificate**.

8. Save your changes.

9. Right-click **RMTSAMPLEFLIGHT**, and select **Activate Service**.

10. Right-click **RMTSAMPLEFLIGHT**, and select **Test Service**.

11. In the browser that opens, select a certificate for authentication.

   If you select the ssodemo certificate, you should see the following:

```
  <app:service>
    <app:collection collection-name="BookingCollection" sap:content-version="1" href="/BookingCollection">
      <app:member title="Flight Booking" sap:member-title="Flight Booking"></app:member>
    </app:collection>
    <app:collection sap:creatable="false" sap:deletable="false" sap:content-version="1" href="/CarrierCollection">
      <app:member title="Carrier" sap:member-title="Carrier"></app:member>
    </app:collection>
    <app:collection sap:creatable="true" sap:searchable="true" sap:content-version="1" href="/TravelAgencyCollection">
      <app:member title="Travel Agency" sap:member-title="Travel Agency"></app:member>
    </app:collection>
    <app:collection sap:searchable="true" sap:content-version="1" href="/TravelAgencies">
      <app:member title="Travel Agencies"></app:member>
    </app:collection>
  </app:service>
</app:service>
```

**i Note**

If the SSL certificate is self-signed, you may see a warning that it is not trusted.
Results

Because you do not have the end-user certificate, you cannot test the authentication process, but you can verify that your system is configured for X.509 authentication.

Task overview: Configure Secure Access to SAP NetWeaver [page 201]

Previous task: Mapping User Certificates in SAP NetWeaver [page 209]

Next task: Creating a Test Application [page 214]

1.4.5.5.1.10 Creating a Test Application

Create an application to test your security configuration.

Prerequisites

Follow the instructions in How to Connect Cloud Platform Mobile Service Account to On-Premise OData Service.

Procedure

2. Select Applications, and click the Create Application icon.
3. Enter these sample values:
   - Application ID – \texttt{ptest}
   - Name – \texttt{ptest}
   - Type – select Hybrid.
   - Description – This is a test app.
   - Vendor – \texttt{ptest}
   - Security Configuration – select Basic.
4. Select Back End, and enter the connection properties:
   - Proxy Type – select OnPremise.
   - Authentication Type – select Principal Propagation.
   - Maximum Connections – 500.
1.4.5.5.1.11 Testing Application Connections to SAP NetWeaver

Test that the application you created can connect to the SAP NetWeaver system.

**Prerequisites**

Install Postman in Chrome.

**Procedure**

1. In a Chrome Web browser, enter `chrome://apps/`, and launch the Postman app.
2. Select **POST** as the method, and enter `https://<host>:<port>/odata/applications/latest/`<ptest>/Connections`, where `<host>` is the name of the machine on which SAP Cloud Platform mobile service for development and operations is running, and `<ptest>` is the application name.
3. Click **Headers**, select **Content-Type**, and enter `application/json` as the value.
5. Click **Send**, and when prompted, enter the user name and password to log in to Mobile Service for Development and Operations cockpit.
6. Copy the value of ApplicationConnectionId.
7. In Postman, enter:
   - The URL of your application, `https://<host>:<port>/ptest/`
   - The value of ApplicationConnectionId
8. Click **Send**. This retrieves data from the back end.
1.4.5.5.1.12 Reviewing SAP Cloud Connector Logs

Review the logs and trace files that SAP cloud connector generates.

Procedure

1. Log in to the SAP cloud connector administration tool.
2. Click Audit, and check the most recent access request. You should see the name of the user who requested the back-end data.
3. Select Log and Trace Files, and select the <date_time>ljs_trace.log file. In the most recent access request, you should see the DN that is being used.
4. (Optional) Copy the text from “backend:” up to, but not including, the “|”. Save the text to a file with a .crt extension, and view it on a Windows computer. You may see something similar to the following:

Certificate Information

Windows does not have enough information to verify this certificate.

Issued to: I808978

Issued by: HCC Intermediate CA

Valid from 7/9/2015 to 7/9/2015

5. Reset the logging level to normal.

Task overview: Configure Secure Access to SAP NetWeaver [page 201]

Previous task: Testing Application Connections to SAP NetWeaver [page 215]

Next task: Configuring Java Back-ends [page 217]
1.4.5.5.1.13 Configuring Java Back-ends

If the back-end system is an SAP Gateway Java instance, edit the instance profile, and set the configuration properties.

Procedure

1. Edit the instance profile \\
usr\sap\<SID>\SYS\profile\<SID>_ <instance>_<host_name>, and set the value of these properties to an asterisk:
   ○ icm/HTTPS/trust_client_with_subject=*  
   ○ icm/HTTPS/trust_client_with_issuer=*  

   In a production system, set the property values appropriate for your system, and restart the system.

2. Log in to Mobile Service for Development and Operations cockpit, and select Configuration SSL.

3. Enable SSL Access Points by adding the server port, and set Client Authentication Mode to Request.

4. Import the Cloud Container issuing certificate that you imported, then restart your system.

5. Modify the application endpoint Java Authentication and Authorization Service (JAAS) stack to use ClientCertLoginModule:
   b. Search for the app by entering a filter value, for example, "security".
   c. Select the app, and click Edit.
   d. Select Authentication Stack, and configure it similar to the following:
Configuring the Internet Communication Manager Certificate

Configure the profile parameters for the Internet Communication Manager (ICM) certificate.

**Task overview:** Configure Secure Access to SAP NetWeaver

**Previous task:** Reviewing SAP Cloud Connector Logs

### 1.4.5.5.1.13.1 Configuring the Internet Communication Manager Certificate

Configure the profile parameters for the Internet Communication Manager (ICM) certificate.

**Procedure**

1. Find the certificate that Cloud Connector uses to issue an end-user certificate.
2. Set the values of the **Subject DN** and the **Issuer** properties, as shown in the image below:

The value of **Subject DN** includes a space between attributes. The value of **Issuer** has no space between attributes.

The state can be represented by any of these characters: S, ST, or SP.

3. To find the correct DN value, increase the trace level for the ICM, and look at the log.
4. Try to access the back-end data via the app. If login fails, decrease the ICM trace level.

5. (Optional) View the `dev_icm` log file.

6. Enter the following values for the ICM parameters:
   - `icm/HTTPS/trust_client_with_issuer` - `CN=HCCRootCA, OU=RIG, O=SAP, L=Greensboro, SP=NC, C=US`
   - `icm/HTTPS/trust_client_with_subject` - `OU=RIG, O=SAP, C=US, SP=NC, CN=HCC, Intermediate CA`

7. Force your changes to take effect by restarting the ICM process.
1.4.6 Maintenance Mode

When runtime Java applications are in maintenance mode and if you log in to the Mobile Service for Development and Operations cockpit, there are different messages you may see.

- If SAP Cloud Platform mobile service for development and operations loads without any message about service status or maintenance, then it is not in maintenance mode.
- If you receive a message that SAP Cloud Platform mobile service for development and operations is currently undergoing maintenance procedures, choose Log On to reload the application and check the maintenance status.
- If the maintenance status cannot be identified, choose Refresh to reload the application. If you select Continue to open, you can access the cockpit, however, the functionality of some features is limited. This practice is not recommended.

1.4.7 Troubleshooting: Common Issues

Overview of common issues.

Disappearing Scroll Bar Issue in Internet Explorer

Problem

After using the browser’s zoom in feature, the scroll bar appears at first, but then disappears after a short time. For example, when viewing the usage reports for a selected application, and using the zoom in feature, the scroll bar first appears at the bottom of the usage window, but then disappears.

Workaround

This is a known issue in Internet Explorer (in a Chrome browser, the scroll bar persists after zooming in). As a workaround, system administrators can use the Microsoft Edge browser available with the Windows 10 operating system.

Error 403 Accessing an Existing Application

Problem

You access an application in the cockpit using an existing registration, and encounter the error message: HTTP Status 403 - The registered user for application connection does not Match the user. This can happen if you update the authentication type associated with the application configuration, and then access the application using an existing registration.
Workaround
The authenticated user name used to access the application must match the authenticated user name that was established during registration. Changing the authentication type can result in a different user name being established after successful authentication. To correct the problem:
1. Delete the existing registration via the cockpit.
2. The client should re-register.

Cannot View Event Logs from the Home Screen

Problem
You cannot view event logs from the Home screen on Internet Explorer 11.

Workaround
You may need to configure Internet Explorer 11 to view the XML event log files:
1. Go to Internet Options > Content > Feeds and Web Slices > Settings.
2. Uncheck “Turn on feed reading view”.

Offline Applications Cannot Connect to Back Ends

Problem
If the back-end server does not support ClientCertificate authentication, offline applications cannot connect.

Workaround
For offline applications, do not specify the keystore location.

Incorrect Password Causes Web Application Registration Problems

Problem
If you enter an incorrect password during client Web application registration, and then enter the correct password in the pop-up challenge dialog, SAP Cloud Platform mobile service for development and operations denies application registration.

Workaround
1. Click Cancel in the challenge dialog.
2. Click OK when you see the error: Username or password incorrect.
3. On the registration page, enter the correct user name and password again. The Web application registers, and then opens.
Not Authorized to Access SAP Cloud Platform mobile service for development and operations

Problem
When attempting to log in to SAP Cloud Platform mobile service for development and operations, you receive the message You are not authorized to access Cloud Platform mobile services. Log in with the correct role, or contact the System Administrator.

Workaround
The user log on account must contain the correct roles before SAP Cloud Platform mobile service for development and operations grants access to users. Ask the account administrator to verify that the account includes an appropriate role.

Error 403 Accessing Mobile Service for Development and Operations cockpit

Problem
If a custom identity provider was configured in SAP Cloud Platform Cockpit, you may get a 403 error when you access Mobile Service for Development and Operations cockpit.

Workaround
1. Verify that the appropriate roles have been granted to the correct user ID. See Setting Up Customer Accounts.

   Note
   The user ID in a role assignment must match the authenticated subject in the SAML Assertion that the IdP returns to SAP Cloud Platform.

2. To troubleshoot the error, install a browser plugin that helps you capture the SAML Request/Response, for example, the SAMLTracer plugin in Firefox or SAML Chrome Panel.
3. Enable and start the trace, and access the Mobile Service for Development and Operations cockpit URL.
4. Examine the SAMLAssertion that the IdP returns to SAP Cloud Platform, specifically the User ID Source in the IdP configuration. See ID Federation with the Corporate Identity Provider.
5. By default, the value of NameID is used as the authenticated user ID. Verify that this value matches the user ID in the role assignment.

Related Information
Set Up Customer Accounts [page 23]
Set Up Customer Accounts [page 23]
1.5 Development

Applications can be developed using REST API or OData SDK.

To develop applications using OData SDK, see Application Development in SAP Mobile Platform and for using SAP Cloud Platform SDK for iOS, see About SAP Cloud Platform SDK for iOS

1.5.1 REST API Application Development Overview

The REST Services, distributed as part of the SAP Cloud Platform mobile service for development and operations, enables standard HTTP client applications running in any platform to leverage mobile platform for security and push features.

Build client applications using third-party developer tools (JavaScript framework and helper libraries), native client libraries, or the libraries provided by the platform OData SDK (iOS and Android platforms only). The mobile platform enables you to manage and monitor the applications, and provides support for native push notification: Apple Push Notification service (APNS), BlackBerry Internet/Enterprise Service (BIS/BES), Google Cloud Messaging (GCM), Windows Notification Service (WNS), or Microsoft Push Notification Service (MPNS).

Application developers must first register an application connection using the REST client and provide the device information, such as device type, password capability, and so on. Once registered, an application can retrieve and update the application connection settings through the REST API. You can enable or disable the push notification only after registering.

Note

You can delete an application connection using the REST API, as long as the application is not in use. Any data that is stored in the custom string of the application connection properties is lost.

During initialization, a client application can download resources (metadata files, multimedia files, and so on), using the resource bundles service. After downloading resources, the application can access OData-compatible data sources through the proxy service, and receive native push notifications triggered by SAP Gateway if push properties are configured and enabled.

This development approach supports:

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

Cloud solutions do not have a Product Availability Matrix (PAM). For more information about cloud solution product versions contact an SAP representative.
Related Information

Set Up Customer Accounts [page 23]
Push Overview [page 35]

1.5.1.1 Configure Applications in Mobile Services Cockpit

Configure an application definition that enables you to manage and monitor the applications using Mobile Service for Development and Operations cockpit.

- **Defining Applications [page 225]**
  Create a new application definition, which enables you to use Mobile Service for Development and Operations cockpit to manage the application.

- **Defining Connectivity [page 227]**
  Define destinations for the selected application.

- **Configuring Form Authentication with Native SAML Providers [page 229]**
  Form authentication uses SAML 2.0 authentication provided by SAP Cloud Platform.

- **Defining Push Notifications [page 229]**
  Configure push-related settings for the selected application.

- **Uploading Client Resources [page 233]**
  Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Cloud Platform mobile service for development and operations. The administrator can modify the client resource bundle settings in Mobile Service for Development and Operations cockpit.

- **Defining Client Log Policy [page 234]**
  Define the policy for uploading client logs to the database.

- **Managing Feature Restriction Policies [page 235]**
  Manage a list of feature restriction policies that apply to all applications from a central location. Feature examples include camera, printer, and push. You can add, allow, restrict, edit, or delete features, and apply changes to existing hybrid applications.

1.5.1.1.1 Defining Applications

Create a new application definition, which enables you to use Mobile Service for Development and Operations cockpit to manage the application.

Context

When you define an application, you can choose a template, or you can just create a shell:
- If you choose one of the templates for a native, hybrid, or Web application, available features for the template are included, including security. Later you can add features that make sense for the application.
- If you choose to create a shell, the security feature is added automatically, and you can add any other features you want.

In a separate step, configure the assigned features, or add more features. See *Managing Application Features* and *Configuring Assigned Features*.

### Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications >> Native/Hybrid** and click **New**.

2. In **New Application**, enter:

<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Config Templates</td>
<td>Leave blank if you want to define a shell application, and add features later. Otherwise select a configuration template:</td>
</tr>
<tr>
<td></td>
<td>○ Native – native applications, including Android, BlackBerry, iOS, Windows Mobile 8, and Windows 8.</td>
</tr>
<tr>
<td></td>
<td>○ Hybrid – Kapsel container-based applications.</td>
</tr>
<tr>
<td></td>
<td>○ Web – applications that run in a Web browser.</td>
</tr>
<tr>
<td></td>
<td>○ Fiori – Kapsel container-based applications that include the Cloud Build feature.</td>
</tr>
<tr>
<td></td>
<td>○ App Modeler - metadata-based applications.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Unique identifier for the application, in reverse-domain notation. This is the application or bundled identifier that is assigned or generated by the application developer. The administrator uses the application ID to register the application with SAP Cloud Platform mobile service for development and operations, and client applications use the Application ID when sending requests to the server. An application ID:</th>
</tr>
</thead>
<tbody>
<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>○ Can contain up to 64 characters</td>
</tr>
<tr>
<td></td>
<td>○ Cannot include spaces</td>
</tr>
<tr>
<td></td>
<td>○ Cannot begin with a period, and cannot contain two consecutive periods</td>
</tr>
<tr>
<td></td>
<td>○ Cannot be any of these case-sensitive keywords: Admin, AdminData, Push, smp_cloud, resource, test-resources, resources, Scheduler, odata, applications, Connections, public, lcm</td>
</tr>
</tbody>
</table>

We recommend that you assign IDs that contain a minimum of two periods, for example, com.sap.mobile.app1.
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Application name can contain only alphanumeric characters, spaces, underscores, and periods, and can be as many as 80 characters long.</td>
</tr>
<tr>
<td>Description</td>
<td>(Optional) The description can contain up to 255 alphanumeric and special characters.</td>
</tr>
<tr>
<td>Vendor</td>
<td>(Optional) Vendor who developed the application. The vendor name can contain only alphanumeric characters, spaces, underscores, and periods, and can be up to 255 characters long.</td>
</tr>
</tbody>
</table>

1. 
2. 
3. Click **Save**. The Info page appears with current settings.

## 1.5.1.1.2 Defining Connectivity

Define destinations for the selected application.

### Context

A destination is a connection to a data source.

SAP Cloud Platform mobile service for development and operations supports one primary endpoint per application ID. However, an administrator can create multiple secondary endpoints for services that an application uses; these secondary endpoints are treated as proxy connections. For applications that access Web services containing relative URLs, add the relative paths to enable the product to handle requests correctly.

In Mobile Service for Development and Operations cockpit, you can view the properties of Fiori applications and connections that were developed using SAP Cloud Platform mobile service for app and device management and imported into SAP Cloud Platform mobile service for development and operations, but you cannot edit their properties; input fields and buttons are disabled or hidden. A default Fiori Client destination, named `com.sap.fiorimobile.con.Fiori_Cloud_Edition`, is created automatically for the Fiori Destination.

### Procedure

1. In Mobile Service for Development and Operations cockpit, select **Mobile Applications** ➤ **Native/Hybrid**
2. Select an application, and then select **Connectivity** under **Assigned Features** (or add it first).
   
   View current mobile, Fiori, and cloud platform destinations for the selected application.
<table>
<thead>
<tr>
<th>Field</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>The destination name.</td>
</tr>
<tr>
<td>Platform Destination Name</td>
<td>For Cloud Platform Destination types, this is the original cloud platform destination name (it may differ from the destination name used in Mobile Service for Development and Operations cockpit). For Mobile Destination types, this is not used (Not Applicable). For Fiori Destination types, this is not used (Not Applicable).</td>
</tr>
<tr>
<td>URL</td>
<td>The destination URL.</td>
</tr>
<tr>
<td>Proxy Type</td>
<td>The proxy type used for the destination - Internet, or On Premise (Cloud Connector).</td>
</tr>
<tr>
<td>SSO Mechanism/Authentication</td>
<td>The single sign-on or authentication security methods employed for the destination.</td>
</tr>
<tr>
<td>Actions</td>
<td>The actions available, such as test an OData application destination, ping a back-end connection, or delete a connection</td>
</tr>
</tbody>
</table>

3. (Optional) Select the new icon ![new_icon] and use the Create Destination dialog to create a new destination. See Creating a Destination for details.

4. (Optional) Select the Add icon ![add_icon] to add an existing destination. Select a destination from the list, and click **OK**.

   **Note**

   If you selected Fiori Destination, a default Fiori Client destination, named com.sap.fiorimobile.con.Fiori_Cloud_Edition, is available by default.

5. Select a row to view its settings in the Destination Overview.
1.5.1.1.3 Configuring Form Authentication with Native SAML Providers

Form authentication uses SAML 2.0 authentication provided by SAP Cloud Platform.

Context

By default, SAP Cloud Platform uses SAP ID service to authenticate users against SAP user accounts and SCN accounts. The SAP Cloud operator configures the native Form/SAML 2.0 at the account level. All applications configured with Form authentication use this native provider. However, each subscriber can further customize an identity provider (IdP) configuration with their own on-premise SAML 2.0 provider in SAP Cloud Platform Cockpit.

Procedure

1. To use an on-premise SAML provider, follow these instructions in the SAP Cloud Platform documentation: ID Federation with the Corporate Identity Provider. To use the native IdP, continue to step 2.
2. In Mobile Service for Development and Operations cockpit:
   a. Select Applications, and for a specific application, select Configure. For Security Configuration, choose Form.

1.5.1.1.4 Defining Push Notifications

Configure push-related settings for the selected application.

The push listener service provided with the server allows back-end systems to send native notifications to devices. Application developers must enable push notification code in applications to use this option.

Android Push Notifications [page 230]
To enable client applications to receive Google Cloud Messaging (GCM) notifications, configure Android push notifications for the selected application.

Apple Push Notifications [page 230]
To enable client applications to receive APNS notifications, configure Apple push notifications for the selected application.

BlackBerry Push Notifications [page 231]
To enable client applications to receive BES/BIS notifications, configure BlackBerry push notifications for the selected application.

Windows Push Notifications [page 232]
To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

Windows Phone Push Notifications [page 232]
To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows phone users running mobile applications, configure Microsoft push notification services (MPNS) for the selected application.

1.5.1.1.4.1 Android Push Notifications

To enable client applications to receive Google Cloud Messaging (GCM) notifications, configure Android push notifications for the selected application.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)
3. Under Android, enter the access key for API key. This is the access key you obtained for your Google API project (https://developers.google.com/cloud-messaging/http-server-ref).
4. Enter a value for Sender ID. This is the project identifier.

1.5.1.1.4.2 Apple Push Notifications

To enable client applications to receive APNS notifications, configure Apple push notifications for the selected application.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications.
2. Select an application, and then select Push Notification under Assigned Features (or add it first)
3. If you do not want to receive APNS push notification, under Apple, select None.
4. To configure APNS for a development and testing environment, select Sandbox; to configure APNS for a production environment, select Production.
   a. Click Browse to navigate to the certificate file.
   b. Select the file, and click Open.
   c. Enter a valid password.
5. Save your changes.
1.5.1.1.4.3 BlackBerry Push Notifications

To enable client applications to receive BES/BIS notifications, configure BlackBerry push notifications for the selected application.

**Prerequisites**

If you intend to use push synchronization with BlackBerry devices, enable push synchronization in the BlackBerry server, using the BlackBerry server documentation.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select *Mobile Applications*.
2. Select an application, and then select *Push Notification* under *Assigned Features* (or add it first).
3. Under Blackberry, select the push type.
   - Select *None* if you do not want to configure Blackberry push notification.
   - Select *BES* to configure Blackberry Enterprise Server (BES) native notification properties.

<table>
<thead>
<tr>
<th>Table 73:</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Server URL</td>
<td>Address in the form \http://&lt;domain_name&gt; or &lt;IP_address&gt;:&lt;port_Number&gt;/pap.</td>
</tr>
<tr>
<td></td>
<td>User Name</td>
<td>(Optional) User who is accessing the URL.</td>
</tr>
<tr>
<td></td>
<td>Password</td>
<td>User password to connect to the URL. If you set a user name, you are required to also enter a password.</td>
</tr>
</tbody>
</table>

- Select BIS to configure Blackberry Internet Server (BIS).

<table>
<thead>
<tr>
<th>Table 74:</th>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Server URL</td>
<td>Address in the form \<a href="https://cp">https://cp</a>&lt;XXXX&gt;.pushapi.eval.blackberry.com/mss/PD_&lt;pushRequest&gt;</td>
</tr>
<tr>
<td></td>
<td>Listener Port</td>
<td>The push listener port for BIS notifications</td>
</tr>
<tr>
<td></td>
<td>Application ID</td>
<td>The unique identifier assigned to the registered push application service</td>
</tr>
<tr>
<td></td>
<td>Password</td>
<td>The configuration property provided by BlackBerry for BIS push.</td>
</tr>
</tbody>
</table>
1.5.1.1.4.4 Windows Push Notifications

To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows desktop and tablet application users, configure Windows push notifications for the selected application.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select *Mobile Applications*.
2. Select an application, and then select *Push Notification* under *Assigned Features* (or add it first)
3. Under *WNS*, enter the application credentials, which are provided by the application developer.

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Package SID</td>
<td>Package security identifier</td>
</tr>
<tr>
<td>Client Secret</td>
<td>Client secret information</td>
</tr>
</tbody>
</table>

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

1.5.1.1.4.5 Windows Phone Push Notifications

To enable the back-end servers connected with SAP Cloud Platform mobile service for development and operations to send toast, tile, badge, and raw updates to Windows phone users running mobile applications, configure Microsoft push notification services (MPNS) for the selected application.

**Context**

*Note*

Only unauthenticated push notification is supported; authenticated push notification for MPNS is not supported.

**Procedure**

1. In Mobile Service for Development and Operations cockpit, select *Mobile Applications*.
2. Select an application, and then select *Push Notification* under *Assigned Features* (or add it first)
3. Under *MPNS*, select *Enable MPNS HTTP Push* to send HTTP push notifications to the device.
4. (Optional) Configure push notifications for each device type supported.

1.5.1.1.5 Uploading Client Resources

Upload client resources, or resource bundles, for the selected application. Resource bundles are containers used by applications to download dynamic configurations, styles, or content from SAP Cloud Platform mobile service for development and operations. The administrator can modify the client resource bundle settings in Mobile Service for Development and Operations cockpit.

Context

Keep in mind these resource bundle guidelines:

- **Supportability** – the resource bundle can be of any type (.pdf, .xls, .xml, or any other extension), with no restrictions.
- **Size** – the resource bundle is restricted in size. The maximum size is 64 MB. For best performance, SAP recommends a maximum size of 1MB. For larger files, work with an application developer for performance issues.
- **Default resource bundle** – the first resource bundle that you upload is considered to be the default. After that, you can upload additional versions of the bundle, but only one can be the default. You can delete obsolete resource bundle versions.
- **URL for the default resource bundle** – https://<hmc base url>/bundles/<ApplicationName>/.
- **URL to access other resource bundles** – https://<hmc base url>/bundles/<ApplicationName>/<BundleName>:<BundleVersion>.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications ➔ Native/Hybrid ➔
2. Select an application, and then select Client Resources under Assigned Features (or add it first)
3. Click the Upload Client Resource icon to create a new Client resource:
   - **Bundle Name** – provide a name to identify the resource.
   - **Version** – provide a version number.
   - **Upload Client Resource** – click Browse, select the file, and confirm.
4. To define a client resource bundle as the default, select it, and click Save.
1.5.1.1.6 Defining Client Log Policy

Define the policy for uploading client logs to the database.

Context

Log settings for an application are applied to all installed instances of an application. You can override those application-level settings for a particular device, by changing the settings for a particular “deviceID”.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Mobile Applications > Native/Hybrid.
2. Select an application, and then select Client Policies under Assigned Features (or add it first).
4. Select the log level (default is None):

<table>
<thead>
<tr>
<th>Log Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Path</td>
<td>For tracing execution flow. Used, for example, in the context of entering and leaving a method, looping, and branching operations. (Not applicable to the offline logging component.)</td>
</tr>
<tr>
<td>Debug</td>
<td>For debugging purposes, includes extensive and low-level information.</td>
</tr>
<tr>
<td>Info</td>
<td>Informational text, used mostly for echoing what has been performed.</td>
</tr>
<tr>
<td>Warn</td>
<td>The application can recover from the anomaly, and fulfill the task, but requires attention from the developer or operator.</td>
</tr>
<tr>
<td>Error</td>
<td>The application can recover from the error, but cannot fulfill the task due to the error.</td>
</tr>
<tr>
<td>Fatal</td>
<td>The application cannot recover from the error, and the severe situation causes fatal termination.</td>
</tr>
</tbody>
</table>
5. Select the time period after which logs are deleted from the database.

You should retrieve these settings at runtime, for more information see Uploading and Viewing Client Logs. By default, log files exist in the database for seven days. To read the logs, see Viewing Event Logs [page 165].
6. Click Save.
1.5.1.1.7 Managing Feature Restriction Policies

Manage a list of feature restriction policies that apply to all applications from a central location. Feature examples include camera, printer, and push. You can add, allow, restrict, edit, or delete features, and apply changes to existing hybrid applications.

Context

From a central location, administrators can manage a list of feature restriction policies for hybrid applications. Each of the centrally maintained feature restriction policies serves as a template. An updated template is automatically applied to new hybrid applications, and can be manually applied to existing ones.

Note

You can override the feature template for individual applications.

Procedure

1. In Mobile Service for Development and Operations cockpit, select Settings Feature Restriction Policy Template.
2. View the current status of feature restrictions.

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Plugin</td>
<td>A list of feature plugins that are available with the application, such as Camera, Calendar, and Push.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique identifier for the plugin.</td>
</tr>
<tr>
<td>Allowed</td>
<td>Indicates whether the feature is allowed or restricted. ○ To allow the feature, select Yes. ○ To restrict the feature, select No.</td>
</tr>
</tbody>
</table>

3. To add a feature restriction policy, click the Add icon.

In the Add Feature Restriction Policy window, enter restriction policy values, and select Save:

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>A unique feature name.</td>
</tr>
<tr>
<td>Plugin</td>
<td>A feature plugin that is available with the application, such as Camera, Calendar, and Push.</td>
</tr>
<tr>
<td>Plugin Name</td>
<td>A plugin that is available with the application, such as Camera, Calendar, and Push.</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Description</td>
<td>A feature plugin description, such as Cordova Camera Plugin, Cordova Contacts Plugin, and SAP Push Plugin.</td>
</tr>
<tr>
<td>JavaScript Module</td>
<td>A comma-separated list of the JavaScript modules that this plugin uses. The JavaScript Module value is the JavaScript API that is used to invoke the plugin, and defined in the plugin.xml file.</td>
</tr>
<tr>
<td>ID</td>
<td>Unique identifier for the plugin. The value comes from the cordova_plugins.js file, which appears in the project after you add a plugin (“pluginId”).</td>
</tr>
<tr>
<td>Allowed</td>
<td>Check to allow the feature.</td>
</tr>
</tbody>
</table>

4. (Optional) To edit a feature restriction policy, click the plugin name.
5. (Optional) To update the feature restriction policies for specific applications, select Update Apps, select the applications, and click Apply.
6. (Optional) To change how the list is presented, click the sort icon.
7. (Optional) To delete a feature plugin, select it, and click the Delete icon. You can select multiple rows for deletion.

1.5.1.2 Application Development using REST API

To access SAP Cloud Platform mobile service for development and operations REST services, develop your HTTP client application to use the REST Services API.

**Authentication Requests [page 237]**
For all requests that require authentication, send the authentication information to SAP Cloud Platform mobile service for development and operations. The credentials, which you provide in the header, depend on the type of security configuration.

**Create Application Connection [page 238]**
You must explicitly register an application connection (also called the registration ID) with mobile platform.

**Create Application Connection with Capability Handling [page 238]**
Starting with version SAP Cloud Platform mobile service for development and operations 1.3 of the connection service, clients can manage form factors and capabilities in the application connections.

**Manage Application Settings [page 239]**
Application settings describe the application connection details such as application ID, security configuration, and customization resource.

**Managing Application Versions Using REST APIs [page 239]**
You can automate application version management by integrating it into your application build or application artifact management processes: use REST APIs to deploy, promote, delete, and view information for application versions.

**SAP Cloud Platform Document Service [page 244]**
The SAP Cloud Platform document service is an implementation of the Content Management Interoperability Services (CMIS) standard, which allows you to create an application-specific repository.
Native Push Notification for a Back End [page 245]
The mobile platform uses the native notification mechanisms provided by individual device platforms such as APNS, GCM, BIS/BES, WNS, and MPNS to send notifications. Back-end systems use the Push REST service to notify the mobile platform about any notification messages it sends to devices.

Registering Clients for Native Push Notifications [page 278]
Enable native push notifications, and register your application to receive push notifications.

Service Document [page 284]
Get the service document for the application connection.

Logging Out Users [page 285]
Terminate an active user session with the user logout service. This is useful for multiple user scenarios.

Storage Service [page 285]
As an application developer, you can use REST services to store device application configuration data in the server.

Client Resources Service [page 285]
As an application developer, you can use REST services to download resources from the server.

Client Log Upload Service [page 286]
As an application developer, you can use REST services to upload client logs in the server.

Push as API Service [page 286]
The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

Document Service [page 286]

Client Usage Report Upload Service [page 286]
The Client Usage Report Upload API allows the application to upload the client usage reports of SAP Cloud Platform mobile service for development and operations.

Role Service [page 286]
As an application developer, you can use REST services to enable mobile devices to retrieve user name and roles associated with the user according to the System for Cross-domain Identity Management (SCIM) protocol. Role service allows you to get the logical roles for the current user, and you can use it to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

1.5.1.2.1 Authentication Requests

For all requests that require authentication, send the authentication information to SAP Cloud Platform mobile service for development and operations. The credentials, which you provide in the header, depend on the type of security configuration.

- 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 URL.
HTTP Header Name: Authorization
HTTP Header Value: Basic <base64 encoded form of username:password>

- External token-based SSO (client acquires SSO token)
  HTTP Header Name: <value provided for 'Client HTTP Values To Send' in the security configuration>
  HTTP Header Value: actual SSO token

- Network-edge token-based SSO (SAP Cloud Platform mobile service for development and operations 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 the certification authority (CA) certificate of the server. The client certificate should be trusted by SAP gateway or any other EIS. You can then use the certificate to register the client and perform the request-response with the server.

1.5.1.2.2 Create Application Connection

You must explicitly register an application connection (also called the registration ID) with mobile platform.

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

Related Information

Create Application Connection [page 294]

1.5.1.2.3 Create Application Connection with Capability Handling

Starting with version SAP Cloud Platform mobile service for development and operations 1.3 of the connection service, clients can manage form factors and capabilities in the application connections.

During registration, the device sends its form factor (such as smartphone or tablet), and the client can send a certain capability name [such as purchaseOrder-display, or a wildcard (*) in case the device has all the capabilities]. When the device user adds or removes a capability, the application connection is updated.
Related Information

Push API Notification Scenarios [page 248]
Push-to-Capability Scenario [page 258]
Create Application Connection with Capability Handling [page 296]

1.5.1.2.4  Manage Application Settings

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

1.5.1.2.5  Managing Application Versions Using REST APIs

You can automate application version management by integrating it into your application build or application artifact management processes: use REST APIs to deploy, promote, delete, and view information for application versions.

Note
You must use a secure port and the HTTPS protocol to make REST API calls. See your client documentation for details about how to submit a request over HTTPS to satisfy the server’s security requirements.

Deploying Hybrid Apps Using the REST API [page 240]
Deploy a new or updated hybrid app to SAP Cloud Platform mobile service for development and operations using the POST application REST API.

Promoting Hybrid Apps Using the REST API [page 241]
Promote a new hybrid app to make it the current version of the application using the PUT application REST API. Only administrators can run this API; developers cannot.

Retrieving Hybrid App Details Using the REST API [page 242]
Retrieve details about a new or current version of a hybrid app using the GET application REST API.

Deleting Hybrid Apps Using the REST API [page 243]
Delete a hybrid app using the DELETE application REST API. Developers can delete only the applications they created.
1.5.1.2.5.1 Deploying Hybrid Apps Using the REST API

Deploy a new or updated hybrid app to SAP Cloud Platform mobile service for development and operations using the POST application REST API.

**Note**
You cannot deploy a hybrid app for a specific platform: everything in the Kapsel application file is deployed. Once the application is deployed, you can promote or delete hybrid apps for specific platforms as needed.

To attach a Kapsel application file as a parameter in a REST client, use the **curl** command line tool.

After an application is deployed, it is considered to be a new version. You can activate it by promoting it to the current version, which allows users to download patches and upgrade the application on their devices.

**Syntax**

Send a POST request to the following URI:

https://<hmc_base_url>/Admin/kapsel/jaxrs/KapselApp/<APP_ID>

**Note**
To authenticate, specify the user name and password in each request.

**Returns**

A response with information about the new and current version of the application. For example:

```json
{"newVersion":
  {"requiredKapselVersion":"1.5",
  "developmentVersion":"1.2.5",
  "description":"An update for the sample app.",
  "revision":-1},
"currentVersion":
  {"requiredKapselVersion":"1.5",
  "developmentVersion":"1.2.4",
  "description":"A sample app.",
  "revision":2}
}
```

If successful, a 201 status code is returned; otherwise, an HTTP failure code and an error message are returned.
Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-server.pem> --X DELETE -i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```

1.5.1.2.5.2 Promoting Hybrid Apps Using the REST API

Promote a new hybrid app to make it the current version of the application using the PUT application REST API. Only administrators can run this API; developers cannot.

Promote a hybrid app for a specific platform or for all platforms.

**i** Note

To authenticate, specify the user name and password in each request.

Syntax

To promote a hybrid app for all platforms, send a PUT request to the following URI:

https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/<APP_ID>

To promote a hybrid app for a specific platform, send a PUT request to the following URI:

https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/<action>/<APP_ID>, where `<action>` is one of:

- stage
- unstage
- promotePending
- promoteStage

After the application is promoted, users can upgrade the application on their devices.

Returns

If successful, a 201 status code is returned; otherwise, an HTTP failure code and an error message are returned.
Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-server.pem> --X DELETE -i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```

1.5.1.2.5.3 Retrieving Hybrid App Details Using the REST API

Retrieve details about a new or current version of a hybrid app using the GET application REST API.

Syntax

Send a GET request to the following URI:

https://<hmc_base_url>/Admin/kapsel/jaxrs/KapselApp/<APP_ID>

<table>
<thead>
<tr>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>To authenticate, specify the user name and password in each request.</td>
</tr>
</tbody>
</table>

Returns

If successful, a 201 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the `curl` command line tool and the `--cacert` flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-server.pem> --X DELETE -i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```
1.5.1.2.5.4 Deleting Hybrid Apps Using the REST API

Delete a hybrid app using the DELETE application REST API. Developers can delete only the applications they created.

Delete a hybrid app from a specific platform or from all platforms.

i Note
To authenticate, specify the user name and password in each request.

Syntax

To delete a hybrid app from all platforms, send a DELETE request to the following URL:

https://<hmc_base_url>/Admin/kapsel/jaxrs/KapselApp/<APP_ID>

To delete a hybrid app from a specific platform, send a DELETE request to the following URL:

https://<host>:<admin_port>/Admin/kapsel/jaxrs/KapselApp/<APP_ID>/<PLATFORM>

To delete applications they created, developers can:

1. Add the X-HTTP-METHOD=DELETE parameter to the request header.
2. Format the request body as this JSON string:

   
   ```json
   [[{"platform": "android","revisions":[1,2,3]}],
   [{"platform": "ios","revisions":[4,5,16]}],
   [{"platform": "android","revisions": [2]},{"platform": "ios"}]
   [{"platform": "android"]
   ```

3. Send a POST request to the following URL:

   https://<hmc_base_url>/Admin/kapsel/jaxrs/KapselApp/delete/<APP_ID>

Returns

If successful, a 201 status code is returned; otherwise, an HTTP failure code and an error message are returned.

Example

This example uses the curl command line tool and the --cacert flag. Your client may require you to pass other arguments or set specific configuration options.

```
curl --user <user>:<password> --cacert <your-server.pem> --X DELETE -i https://localhost:8083/Admin/kapsel/jaxrs/KapselApp/MyTestAppId
```
1.5.1.2.6 SAP Cloud Platform Document Service

The SAP Cloud Platform document service is an implementation of the Content Management Interoperability Services (CMIS) standard, which allows you to create an application-specific repository.

Mobile service for development and operations acts as a bridge for mobile applications to access the SAP Cloud Platform document service through the js library or any other client library that is compatible with the CMIS 1.1, for example, see Apache Chemistry Parts.

SAP Cloud Platform mobile service for development and operations creates a default CMIS repository, and tenants have their own space within this repository. For example, to add a new client application in Mobile Service for Development and Operations cockpit, a directory with same name is created under the server repository. Each registered user of the client application can create a document or subdirectory under his or her own application directory. If you use Mobile Service for Development and Operations cockpit to delete a client application, the relative directory, subdirectory, and documents are also deleted automatically.

The document service is accessible to device-application developers and administrators.

Device Application Developer

To access the document service, each device application must register with SAP Cloud Platform mobile service for development and operations by creating an application connection. The URL to the application folder is: https://<HMC base URL>/hanamobile-hmtenant1.neo.ondemand.com/mobileservices/persistence/v1/json/<appid>/$metadata

The registered application cannot:

- Access other application folders or the root folder
- Query repository information
- Support CMIS query language for the device application

**Note**

By default, all subfolders and documents created under the application folder are accessible to all registered application users. To set a folder or a document as private, clear Access Control Entries (ACE).

To create multiple versions of a document, set `objectTypeId: sap:versioned` in place of `file` as shown in the following example:

```javascript
var file = $("input[name='file']")[0].files[0]; // Assume file is an input file
var input = {
    "cmis:name": "test-file.txt",
    "cmis:objectTypeId": "sap:versioned",
};

session.createDocument(folderId, file, input, "text/plain", "major", null, null, {
    request : {
        success : function(data) {
        },
        error : function(e) {
        }
    }
});
```
Administrator

The SAP Cloud Platform mobile service for development and operations administrator has full access rights to the root folder and to all application folders. By default, all mobile service administrators with the {sap:builtin}admin role can access the document service.

The URL to the application folder is: https://<HMC base URL>/hanamobile-hmtenant1.neo.ondemand.com/mobileservices/persistence/v1/json/$metadata.

1.5.1.2.7 Native Push Notification for a Back End

The mobile platform uses the native notification mechanisms provided by individual device platforms such as APNS, GCM, BIS/BES, WNS, and MPNS to send notifications. Back-end systems use the Push REST service to notify the mobile platform about any notification messages it sends to devices.

Request

**URL:** https://<HMC base URL>/restnotification/<registration ID>

**HTTP Method:** POST

Request Parameters

Table 78:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>restnotification</td>
<td>Mandatory</td>
<td>Received from the proxy push end-point.</td>
</tr>
<tr>
<td>registration ID</td>
<td>Mandatory</td>
<td>Is sent to the device when a user registers and connects to the application from the device.</td>
</tr>
</tbody>
</table>
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gcmNotification</td>
<td>Device specific</td>
<td>Used for communication between SAP Cloud Platform mobile service for development and operations and the GCM server, but provides notification elements used to send GCM push requests to SAP Cloud Platform mobile service for development and operations.</td>
</tr>
<tr>
<td>wnsNotification</td>
<td>Device specific</td>
<td>Used for communication between SAP Cloud Platform mobile service for development and operations and the WNS server, but provides notification elements used to send WNS push requests to SAP Cloud Platform mobile service for development and operations.</td>
</tr>
</tbody>
</table>

You can also send notification data using URL arguments.

- **Notification Data Sent Using Push API [page 247]**
  A push message appears as a notification on a device, informing the user of an action he or she must take. In order to send push notifications to an application, you must have Notification User privilege assigned to your user ID.

- **Push Notification JSON Payload Handling [page 261]**
  SAP Cloud Platform mobile service for development and operations provide push notification JSON payload handling for APNS, GCM, and WNS. When enabled, the mobile platform maps JSON payload values to custom platform values.

- **Notification Data Sent Through HTTP Headers [page 272]**
  Notification data can be sent by the back end as generic HTTP headers or as device platform-specific HTTP headers.

- **SAP Gateway Notification Support [page 274]**
  There are no specific handling requirements for sending notifications on the SAP Gateway side. mobile service for development and operations sends notifications using gateway-specific headers.

- **Notification Sent in URL Format [page 276]**
  Notification data can also be sent by using the REST client, using URL arguments as part of the mobile platform push endpoint, or as the delivery address URL.

### Related Information

- Set Up Customer Accounts [page 23]
- Notification Sent in URL Format [page 276]
- Notification Data Sent Using Push API [page 247]
1.5.1.2.7.1 Notification Data Sent Using Push API

A push message appears as a notification on a device, informing the user of an action he or she must take. In order to send push notifications to an application, you must have Notification User privilege assigned to your user ID.

Configuring Authentication Provider Settings in Mobile Services Cockpit (SAP Cloud Platform)

By default, SAP Cloud Platform uses SAP Cloud ID service to authenticate users against SAP user accounts. Assign the Notification User role to the SAP user ID to be able to send the push notification to the device.

1. Open the SAP Cloud Platform Cockpit
2. Select Services.
3. Under mobile service for development and operations, select Roles.
4. Select the Notification User role and assign user ID to the role.

**Note**

In Cloud Platform mobile Services, an unauthenticated user is referred as a public user rather than as a nossec_identity user as in the on-premise version of the server.

Testing Notification Service

You can use any REST tool, such as “Advanced Rest Client” or “Postman”, available from the Google Chrome Web store for testing.

The restnotification API sends native push notifications to the applications. This RESTful service provides more flexibility for sending push messages than existing interfaces that are based on HTTP headers or URL parameters. Earlier push interfaces required that you send messages to a registration ID. The restnotification interface also sends the message to a specific user or to all users of a specific application.

The restnotification API sends messages to multiple recipients. The messages are queued in the server and sent out asynchronously.

**Request**

**URL:** http[s]://<HMC base URL>/restnotification/<resource>

**HTTP Method:** GET

Push API Notification Scenarios [page 248]
Send push notifications to devices that are registered to an application.

**Push-to-Capability Scenario [page 258]**

The push-to-capability scenario is a push notification variation. This scenario enables you to push notifications to applications that have certain capabilities rather than to individual applications.

**Related Information**

Native Push Notification for a Back End [page 245]

### 1.5.1.2.7.1.1 Push API Notification Scenarios

Send push notifications to devices that are registered to an application.

**Request**

**URL:** https://<HMC base URL>/restnotification/application/<applicationId>

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application.</td>
</tr>
</tbody>
</table>

**Request Body Example**

```bash
> POST /restnotification/application/123456789 HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {
>   "alert": "alertval",
>   "badge": 1,
>   "data": "testData",
>   "sound": "soundval"
> }
< HTTP/1.1 201 Created
< Content-Length: 0
< Date: Mon, 05 May 2014 00:29:38 GMT
< Server: SAP
```

In this scenario, a status code 201 indicates that the server accepts the push notification request. The server forwards these requests to the external push service such as GCM, BES, BIS, APNS, WNS and so on. The status code does not indicate that the server has successfully delivered the notification to the devices.
Response

Other possible HTTP status codes, you may encounter:

Table 80:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>400 Bad Request</td>
<td>The request is invalid. Verify the request body.</td>
</tr>
<tr>
<td>401 Forbidden</td>
<td>The user who issued the request does not have the required privileges. Ensure that the user is assigned to the Notification User role.</td>
</tr>
<tr>
<td>403 Authentication required</td>
<td>No or incorrect credentials provided. Enter the correct credentials.</td>
</tr>
</tbody>
</table>

Users and Devices

To send push notification to all the devices registered to a particular user, use:

**URL:** http[s]://<HMC base URL>/restnotification/application/<applicationId>/user/<userID>

**Request Body Example**

```plaintext
> POST /restnotification/application/123456789/user/timmitester HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {
>   "alert": "alertval",
>   "badge": 1,
>   "data": "testData",
>   "sound": "soundval"
> }
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIDSSO=C05E58BE3CFC685ABB945D53C2AF14FD; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSID=4CC5BC2943E5D3A9B5D924888FC28CB060034F0092911A66B9F079047077798C; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:32:35 GMT
< Server: SAP
```

Registration ID

To send push notification to a device by using an application registration ID, use:

**URL:** http[s]://<HMC base URL>/restnotification/registration/<applicationRegistrationId>
**Request Body Example**

```shell
> POST /restnotification/registration/9f847e51-3242-4899-9193-39b6e840d657
HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 127
> {  
>     "alert": "alertval",
>     "badge": 1,
>     "data": "testData",
>     "sound": "soundval"
> }
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIDSSO=D541E4898166AB304F506D13C0C3F1D0; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSID=FDB39F98A6E1AD4373765A58E094A14B8FDFA289CC70E51B77A284C50736; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:36:15 GMT
< Server: SAP
```

**Users per Application**

To send push notification to all the users of an application, use:

**URL:** `http[s]://<HMC base URL>/restnotification/application/<applicationId>/user`

**Request Body Example**

```shell
POST /restnotification/application/123456789/user HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 277
> {  
>     "notification": {  
>         "alert": "alertval",
>         "badge": 1,
>         "data": "testData",
>         "sound": "soundval"
>     },
>     "users": [  
>         "timmitester",
>         "user1",
>         "user2"
>     ]
> }
< HTTP/1.1 201 Created
< Set-Cookie: X-SMP-SESSIDSSO=9AD06173C8AB9FC05FD6A8DC5BB9AE; Path=/; HttpOnly
< Set-Cookie: X-SMP-SESSID=DFB2D2AC4EBAA4903553EB7C5A0C90870BD4B8F3A3DC19A5FD984673EB1BD646; Path=/; HttpOnly
< Content-Length: 0
< Date: Mon, 05 May 2014 00:38:32 GMT
< Server: SAP
```
Registration List

To send push notifications to a list of registrations, use:

**URL:** [http[s]://<HMC base URL>/restnotification/registration/](http[s]://<HMC base URL>/restnotification/registration/)

**Request Body Example**

```bash
POST /restnotification/registration HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> Content-Length: 466
> {
>   "notification": {
>     "alert": "alertval",
>     "badge": 1,
>     "data": "testData",
>     "sound": "soundval"
>   },
>   "registrations": [
>     "3078e166-f144-4288-9dbc-1d192afe18d8",
>     "9f847e51-3242-4899-9193-39b6e8d0d657",
>     "4d1ccdf9-058a-42cf-a625-c4ed48944729",
>     "f05dc905-b859-45fa-afdc-da3b630d2b48",
>     "282be579-783e-40fb-b376-25bed5e13606"
>   ]
> }
> < HTTP/1.1 201 Created
> < Set-Cookie: X-SMP-SESSIDSSO=BCA5FCB41DD7F451410E3E8BB59E8F7A; Path=/; HttpOnly
> < Set-Cookie: X-SMP-SESSID=2AC74022B258178ED3A88E4B2FA10AB41093F53C3D0A77976FE6F076F1E3CC2; Path=/; HttpOnly
> < Content-Length: 0
> < Date: Mon, 05 May 2014 00:41:52 GMT
> < Server: SAP
```

Capability

Use capability to identify device capabilities. This enables you to push notifications to applications with specific capabilities rather than to individual applications. To send push notifications to applications that support specific capabilities use:

**URL:** [http[s]://<HMC base URL>/restnotification/capability/<capabilityName>/](http[s]://<HMC base URL>/restnotification/capability/<capabilityName>/)

Capability supports two modes:

- **Wildcard (★):** the device has all capabilities. When a push notification is sent, the device form factor must match.
  
  For example, Jean registers a device with a wildcard capability `capabilityName: *` and form factor: `tablet`, and Jake registers with `capabilityName: *` and form factor: `phone`. When the notification capability: 'purchaseOrder-display' and form factor: `phone` is pushed to both users, only Jake gets the notification. Jean does not get the notification, because the form factor does not match.

- **Match capability name only:** the device has a certain capability name. When a push notification is sent, the notification must match the capability, and the form factor is ignored.
For example, Yijie registers a device with a specific capability name `capability: 'purchaseOrder-display'` and form factor: `phone`. When a notification is pushed to capability: `capability: 'purchaseOrder-display'` and form factor: `tablet`, Yijie receives the notification because the capability matches. The form factor `formFactor: tablet` is ignored.

**Note**

You can use `CapabilityName` either as a wildcard (*), or as specific strings, but not as a string + wildcard (*), such as `purchase*`. For example, if you set `CapabilityName=purchase*` using the REST client, and then send a notification to `purchaseOrder-display`, the device does not get the notification.

**Request Body Example**

```http
POST http://localhost:8082/restnotification/capability/display HTTP/1.1
> Accept: application/json
> Authorization: Basic cHVsaDpzZWNyZXQ=
>
{
  "notification": {
    "data": "{\"NotificationId\":\"005056AB5B8D1ED4B99CC017A78D2429\",\"Text\":\"You have a new purchase order for approval\",\"NavigationTargetObject\":\"purchaseOrder\",\"NavigationTargetAction\":\"display\",\"NavigationTargetParam\":\"{\"ID\":\"4711\"}\",\"Actions\":\[\{"ActionId\":\"approve\",\"ActionText\":\"Approve\",\"BulkActionText\":\"Approve all\",\"Nature\":\"POSITIVE\"\},\{"ActionId\":\"reject\",\"ActionText\":\"Reject\",\"BulkActionText\":\"Reject all\",\"Nature\":\"NEGATIVE\"\}\],\"NotificationTypeId\":\"purchaseOrder\"},
    "alert": \"You have a new purchase order for approval\",
    "sound": \"beep\",
    "customParameters": {
      "apns.category": \"INVITE_CATEGORY\"
    }
  },
  "users": [
    {"badge": 3, "formFactor": ["tablet", "smartphone"], "user": "john"},
    {"badge": 2, "formFactor": ["smartphone"], "user": "jane"}
  ]
}< HTTP/1.1 201 Created
< Content-Type: application/json
```

**Response Body**

```json
>
{
  "status": {
    "value": "OK",
    "code": 0
  },
  "results": [
    {
      "status": {
        "value": "OK",
        "code": 0
      }
    }
  ]
}
```
Capability (Enhanced Badge Handling)

There is not always a one-to-one relationship between the capability that is pushed to, and the badge number for the relevant mobile app. A single mobile app could be capable of handling multiple different capabilities. However, this information is only known by SAP Cloud Platform mobile service for development and operations, not by the back end. Consequently, the back end cannot deliver the accurate badge number for a random mobile application, it can only deliver badge numbers per capability. Typically the API only sends a single badge number, which is too restrictive for this scenario, and may result in inaccurate badge numbers showing up on a user’s device.

A single push would still be targeted towards a particular capability, but the capability-based push API has been enhanced so that the back end can send the badge number per capability. SAP Cloud Platform mobile service for development and operations calculates the actual badge number for an application based on its registered capabilities, and sends the sum of all individual badge values to the device.

POST URL:
https://<hmc base URL>/restnotification/capability/capability2

Payload:
```json
{
  "notification": {
    "data": "...",
    "alert": "You have a new purchase order for approval",
    "customParameters": {
      "apns.category": "action"
    }
  },
  "users": [
    {
      "badge": 3,
      "formFactor": ["tablet"],
      "user": "jean"
    },
    {
      "badges": {
        "capability1": 2,
        "capability2": 4,
        "capability3": 8
      },
      "formFactor": ["phone"],
      "user": "jake"
    }
  ]
}
```
The badge property can either be a plain number, which is sent as is to the native push service (the default implementation); or alternatively the badges property can contain a map of capability/badge pairs (the enhanced implementation). The actual badge value is then determined based on the target application’s registered capabilities.

In the example above, the push is delivered to the mobile apps that "jake" has installed, which have registered for "capability2". Assuming "jake" has one app with the wildcard capability, and one app with capabilities "capability2" and "capability3"; the badge count for the app with wildcard capability would be 14, the badge count for the other app would be 12.

The API enables a badge to be reset without sending an actual notification, and allows the notification data and alert to be empty. In cases where capability context is not provided, the API supports the wildcard "*" as a capability in the URL. In other words, a push to/restnotification/capability/* must be delivered to all apps for a user that has registered with any capability (or the wildcard capability), and all other matching rules must be obeyed, for example, the form factor must still match.

Customize Push Notification Types

Use customParameters to override the value for a particular notification type. Customize push notification types-alert, badge, data, and sound in the payload by prefixing with:

- apns
- bbbis
- bbbes
- gcm
- mpns
- wns

**HTTP Method: POST**

**Example:** Push to all users to the application with application ID "XYZ". Issue a POST method on:

```plaintext
> POST /restnotification/application/XYZ HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json;charset=utf-8
> 
> { 
>   "alert": "alertval",
>   "badge": 1,
>   "data": "testData",
>   "sound": "soundval"
> }
```

**Example:** To reset or override the value of the notification type parameter - sound in your Android device, you can use the customParameters to override the value of the sound parameter:

```plaintext
> {
>   "alert": "alertval",
>   "badge": 1,
>   "customParameters": {
```
Category (APNS)

Use `category` for "actionable" APNS push notifications. These notifications can be sent through SAP Mobile Platform and SAP Cloud Platform mobile service for development and operations directly, or through Push Hub.

Example: REST(ful) pushPush request containing a JSON payload. The category is a sub-element of the customParameters element called "apns.category". Issue a POST method on:

```bash
> POST /restnotification/registration/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/json
> Authorization: Basic cHVzaDpzZWNyZXQ=
> Content-Length: 117
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
> {
>   "alert": "PushAlert",
>   "data": "pushTest",
>   "customParameters":
>   {"apns.category":"SoapUICategory" }
> }
```

Example: non-SAP Gateway notification. Include the header "X-SMP-APNS-CATEGORY". Issue a POST method on:

```bash
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> X-SMP-APNS-CATEGORY: SoapUICategory
> X-SMP-APNS-DATA: pushTest
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: SAP Gateway notification Include the header "X-SAP-POKE-CATEGORY". Issue a POST method on:

```bash
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip,deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> X-SAP-POKE-DATA: pushTest
> X-SAP-POKE-CATEGORY: SoapUICategory
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```
Example: URL parameter encoded. Append the parameter "category=" to the request URL. Issue a POST method on:

```text
> POST /Notification/<applicationId>?alert=PushAlert&data=pushTest&category=SoapUICategory HTTP/1.1
> Accept-Encoding: gzip, deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXM=
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

### Content Available (APNS)

Use `contentAvailable` for "actionable" APNS push notifications, which enable users to take action without changing focus. These notifications can be sent through SAP Mobile Platform and SAP Cloud Platform mobile service for development and operations directly, or through Push Hub.

Example: REST(ful) push request containing a JSON payload. The `content-available` field is a sub-element of the `customParameters` element called "apns.contentAvailable", and is of type "boolean". Issue a POST method on:

```text
> POST /restnotification/registration/<applicationId> HTTP/1.1
> Accept-Encoding: gzip, deflate
> Content-Type: application/json
> Authorization: Basic cHVzaDpzZWNyZXM=
> Content-Length: 146
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)

{ 
  "data": "pushTest",
  "customParameters":
  {"apns.contentAvailable":"true" }
}
```

Example: non-SAP Gateway notification. Include the header "X-SMP-APNS-CONTENT-AVAILABLE". Issue a POST method on:

```text
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip, deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXM=
> X-SMP-APNS-CONTENT-AVAILABLE: true
> X-SMP-APNS-DATA: pushTest
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

Example: SAP Gateway notification Include the header "X-SAP-POKE-CONTENT-AVAILABLE". Issue a POST method on:

```text
> POST /Notification/<applicationId> HTTP/1.1
> Accept-Encoding: gzip, deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXM=
```
Example: URL parameter encoded. Append the parameter "Content_Available=" to the request URL. Issue a POST method on:

```bash
> POST /Notification/<applicationId>?
alert=PushAlert&data=pushTest&Content_Available=true HTTP/1.1
> Accept-Encoding: gzip, deflate
> Content-Type: application/xml
> Authorization: Basic cHVzaDpzZWNyZXQ=
> Content-Length: 0
> Host: localhost:8080
> Connection: Keep-Alive
> User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

### Sound (APNS)

Use the `sound` to submit the name of the sound file to be played along with a push message when using the gateway header. This header can hold an arbitrary string representing the name of the sound file, which is appended to the APNS message's sound parameter during the process of building the APNS payload. To play the default sound the user has configured for notifications, set the parameter to the value "default": "sound": "default".

Example: the RESTful SAP Gateway notification includes the "X-SAP-POKE-SOUND" header in the request.

```bash
POST http://localhost:8080/Notification/fa8ed84c-9033-45c0-a0a0-971a2d217367 HTTP/1.1
Accept-Encoding: gzip, deflate
Content-Type: application/xml
Authorization: Basic ****
X-SAP-POKE-SOUND: SomeSoundFile
X-SAP-POKE-TITLE: Alert from SMP3
Content-Length: 0
Host: localhost:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

#### Request Body Example

```json
> POST /restnotification/application/123456789 HTTP/1.1
> Authorization: Basic cHVzaDpzZWNyZXQ=
> User-Agent: curl/7.36.0
> Host: localhost:8080
> Accept: */*
> Content-Type: application/json; charset=utf-8
> Content-Length: 127
> {
>     "alert": "alertval",
>     "badge": 1,
>     "data": "testData",
>     "sound": "soundval"
> }
< HTTP/1.1 201 Created
< Content-Length: 0
```
In this scenario, the "soundval" file is associated with the request.

Example outgoing payload to APNS:

```
{"aps":{"alert":"Alert from SMP3","sound":"SomeSoundFile"}}
```

Related Information

Push-to-Capability Scenario [page 258]
Create Application Connection with Capability Handling [page 296]
Create Application Connection with Capability Handling [page 238]

1.5.1.2.7.1.2 Push-to-Capability Scenario

The push-to-capability scenario is a push notification variation. This scenario enables you to push notifications to applications that have certain capabilities rather than to individual applications.

Applications provide capability information, such as 'purchaseOrder-display', when they register an application connection. The platform then uses the capability information to push notifications to the device.

Usage

Two supported modes:

- Wildcard (*): the device has all capabilities; notifications must match the device's form factor.
- Match capability name only: the device has a certain capability name; notifications must match the capability, and the form factor is ignored.
  
Enhanced badge handling is supported as part of capabilities. The mobile app handles multiple different capabilities, and the sum of all individual badge values is sent to the device.

Wildcard (*)

A wildcard indicates that the device has all capabilities. When someone sends a notification to a certain capability name, then the device form factor must match.

For example, Jean registers a device with a wildcard capability `capabilityName: *` and `form factor: tablet`, and Jake registers with `capabilityName: *` and `form factor: phone`. When the notification capability: 'purchaseOrder-display' and `form factor: phone` is pushed to both users, only Jake gets the notification. Jean does not get the notification, because the form factor does not match.

**POST URL:**

http://smpqa-win12-01.oak.sap.corp:8080/restnotification/capability/
purchaseOrder-display

Payload:
**Match Capability Name Only**

A specific capability name indicates that the notification must match the capability, and that the form factor is ignored.

For example, Yijie registers a device with a specific capability name `capability: 'purchaseOrder-display'` and form factor: phone. When a notification is pushed to capability: `purchaseOrder-display` and formFactor: tablet, Yijie receives the notification because the capability matches. The form factor `formFactor: tablet` is ignored.

**Note**

You can use `CapabilityName` either as a wildcard (*), or as specific strings, but not as a string + wildcard (*), such as `purchase*`. For example, if you set `CapabilityName=purchase*` using the REST client, and then send a notification to `purchaseOrder-display`, the device does not get the notification.

**Enhanced Badge Handling**

There is not always a one-to-one relationship between the capability that is pushed to, and the badge number for, the relevant mobile app. A single mobile app could handle multiple different capabilities. However, this information is only known by SAP Cloud Platform mobile service for development and operations, and not by the back end. Consequently, the back end cannot deliver the accurate badge number for a random mobile application; it can only deliver badge numbers per capability. Typically, the API sends only a single badge number, which is too restrictive for this scenario, and may result in inaccurate badge numbers showing up on a user’s device.

A single push is still targeted toward a particular capability, but the capability-based push API has been enhanced so that the back end can send the badge number per capability. SAP Cloud Platform mobile service for development and operations calculates the actual badge number for an application based on its registered capabilities, and sends the sum of all individual badge values to the device.

**POST URL:**

https://<hmc base URL>/restnotification/capability/capability2
The badge property can either be a plain number, which is sent as-is to the native push service (the default implementation); or alternatively, the badges property can contain a map of capability/badge pairs (the enhanced implementation). The actual badge value is then determined based on the target application’s registered capabilities.

In the example above, the push is delivered to the mobile apps that “jake” has installed, which have registered for “capability2”. Assuming “jake” has one app with the wildcard capability, and one app with “capability2” and “capability3”: the badge count for the app with wildcard capability is 14, the badge count for the other app is 12.

The API enables a badge to be reset without sending an actual notification, and allows the notification data and alert to be empty. If capability context is not provided, the API supports the wildcard “*” as a capability in the URL. In other words, a push to /restnotification/capability/* must be delivered to all apps for a user that has registered with any capability (or the wildcard capability), and all other matching rules must be obeyed, for example, the form factor must still match.

Related Information

Push API Notification Scenarios [page 248]
Create Application Connection with Capability Handling [page 296]
Create Application Connection with Capability Handling [page 238]
1.5.1.2.7.2 Push Notification JSON Payload Handling

SAP Cloud Platform mobile service for development and operations provide push notification JSON payload handling for APNS, GCM, and WNS. When enabled, the mobile platform maps JSON payload values to custom platform values.

**APNS Push Notification Payload Handling [page 261]**

APNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices. The mobile platform maps the incoming JSON payload to APNS custom values.

**GCM Push Notification Payload Handling [page 264]**

GCM provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

**WNS Push Notification Payload Handling [page 269]**

WNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

1.5.1.2.7.2.1 APNS Push Notification Payload Handling

APNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices. The mobile platform maps the incoming JSON payload to APNS custom values.

**Payload Handling for APNS**

This section provides examples of JSON payload handling for APNS custom values.

- **data**
  
  The mobile platform maps `data` to an APNS custom value using `key=data`. The value must be a String. For example, the client sends:
  
  ```
  { "data": "{"acme1":"bar", "acme2" : ["bang", "whiz"]}"
  }
  ```

  The mobile platform forwards this JSON to APNS:

  ```
  { "data": "{"acme1":"bar", "acme2" : ["bang", "whiz"]}"
  }
  ```

- **alert**
  
  The mobile platform maps `alert` to NS custom value `aps.alert`. The value of `alert` can be either a String, or a JSON Object converted to a String.

  - **alert Value as a String**
    
    The mobile platform sets `alert` as value of `aps.alert`.

  - **alert Value as a JSON Object Converted to a String**
The mobile platform parses the String and converts it to a JSON Object. It supports all keys currently supported, such as `body`, `title-loc-key`, `title-loc-args`, `action-loc-key`, `loc-args`, `launch-image`.

**Note**

The key `body` has a higher priority than `alter_body`. If `body` is included, then `alter_body` is ignored. If only `alter_body` is included, it is used.


For example:

```json
{  
  "alert": "{"alert_body":"ignoreThis\\","body\\":"Your vacation request has been approved.\\","action-loc-key": "\"VIEW\\","loc-key": "GAME_PLAY_REQUEST_FORMAT\\", "loc-args": [ "Jenna\\","Frank\\"], "launch-image": "push_icon.png\\","title": "GameRequest\\","title-loc-key": "titleLocKeyVal\\", "title-loc-args": ["foo\\","bar\\"]}"}
```

is forwarded as:

```json
{
  "aps": {
    "alert": {
      "body": "Your vacation request has been approved.\\", "title": "Game Request\\", "launch-image": "push_icon.png\\", 
      "loc-args": [ "Jenna\\", "Frank\\" ],
      "title-loc-args": [ "foo\\", "bar\\" ],
      "action-loc-key": "VIEW\\", "loc-key": "GAME_PLAY_REQUEST_FORMAT\\", "title-loc-key": "titleLocKeyVal\\"
    }
  }
}
```

**sound**

The mobile platform maps `sound` to the APNS custom value `aps.sound`. The value must be a String. For example:

```json
{  
  "sound": "chime.aiff"  
}
```

is forwarded as:

```json
{  
  "aps": {  
    "sound": "chime.aiff"  
  }  
}
```

**badge**
The mobile platform maps `badge` to the APNS custom value `aps.badge`. The value must be a non-negative Number. For example:

```json
{ "badge": 1 }
```

is forwarded as:

```json
{ "aps": { "badge": 1 } }
```

**apns.category (as a child of "customParameters")**

The mobile platform maps `apns.category` to the APNS custom value `aps.category`. The value must be String. When the client sends the following payload:

```json
{ "alert": "New vacation request", "customParameters": { "apns.category": "NEW_MESSAGE_CATEGORY" } }
```

it is forwarded as:

```json
{ "aps": { "category": "NEW_MESSAGE_CATEGORY", "alert": "New vacation request" } }
```

**apns.contentAvailable (as a child of "customParameters")**

The mobile platform maps `apns.contentAvailable` to the APNS custom value `aps.contentAvailable`. The value must be either 0 or 1. When the client sends the following payload:

```json
{ "customParameters": { "apns.contentAvailable": true } }
```

it is forwarded as:

```json
{"aps":{"content-available":1}} - "apns.customValues" as child from "customParameters"
```

The mobile platform maps it to APNS custom values. The value must be a JSON Object converted to String. For example, when the client sends the following payload:

```json
{ "alert": "New vacation request", "customParameters": { "apns.customValues": "{"acme1":"bar", "acme2" : ["bang", "whiz"]}" } }
```

it is forwarded as:

```json
{ "aps": { "alert": "New vacation request"}, "acme2": [ "bang", "whiz" ], "acme1": "bar" }
```
1.5.1.2.7.2.2 GCM Push Notification Payload Handling

GCM provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

Payload Handling for GCM

This provides information and examples of JSON payload handling for GCM custom values. Notification elements send GCM push requests to the mobile platform.

JSON Payload Support for GCM Push Notifications

When sending push notifications to GCM, JSON payload handling is transparent to the client since it affects only the communication between the mobile platform and the GCM server. Use these notification elements to send GCM push requests to the mobile platform:

- **priority** - allows the definition of a priority that is forwarded to the respective notification type handler. In case of GCM, the values “normal”, which is the default if the field is left blank, and “high” are valid values.
- **gcmNotification** - holds GCM-specific notification elements. Unless specified, the data type for each element is “string”. These are optional.
  - collapseKey
  - delayWhileIdle - holds a Boolean value
  - timeToLive - holds a long value (TTL in milliseconds)
  - restrictedPackageName
  - title - if set, this field takes priority over the “alert” field of the notification element
  - body
  - icon
  - sound - if set, this field takes priority over the “sound” field of the notification element
  - tag
  - color
  - clickAction
  - bodyLocKey
  - bodyLocArgs - must hold a serialized JSON list of values
  - titleLocKey
  - titleLocArgs - must hold a serialized JSON list of values

For more detailed descriptions of the GCM specific fields, see GCM Connection Server Reference > HTTP Protocol, Downstream HTTP messages (JSON): https://developers.google.com/cloud-messaging/http-server-ref#send-downstream.

If the JSON payload handling for GCM custom values feature is inactive, the old GCM notification handling behavior is applied, meaning that the above-mentioned notification elements are not part of the push message sent to GCM, and the message sent to GCM remains in plain-text format, instead of JSON format. Through the feature flag, the mobile platform GCM notification handler stays backward compatible, offering the possibility of including the (old) notification payload via headers and URL parameters.
The following example shows a complete push request to the mobile platform, using every element available:

```json
{
    "alert": "Updates Available",
    "badge": 1,
    "data": "{'version':'1.13','size':'14MB'}",
    "priority": "high",
    "sound": "DefaultNotificationSound",
    "gcmNotification": {
        "title": "The Title For The App",
        "icon": "TheIcon",
        "body": "The Notification Body",
        "sound": "OverrideSound",
        "color": "Blue",
        "tag": "TheTag",
        "collapseKey": "TheCollapseKey",
        "delayWhileIdle": true,
        "timeToLive": 10,
        "restrictedPackageNames": ["com.sap.test"],
        "clickAction": "TheClickAction",
        "bodyLocKey": "message",
        "bodyLocArgs": ["msg1", "msg2"],
        "titleLocKey": "titleMessage",
        "titleLocArgs": ["tmsg1", "tmsg2"]
    },
    "customParameters": {
        "gcm.badge": 2
    }
}
```

**Notification Element Validation**

In earlier versions, only very basic notification element validation was applied, verifying only that at least one of the following elements "alert", "badge" or "data" contained information.

This behavior has been changed to accommodate the "send-to-sync" functionality of GCM, and to provide more granular methods of validating notification payload that is tailored to a specific notification type. See Send-to-Sync, below.

The input validation is still performed synchronously, meaning a notification with invalid values is not accepted, and results in an immediate error response being returned. Summary of the latest changes to the mobile platform GCM notification handler:

- **timeToLive** - validation ensures that the value of the field is in the range of 0 to 2419200.
- **priority** - validation ensures that the field contains either the value "normal" or "high", or no value at all, for which GCM assumes the default value of "normal".
- **bodyLocArgs** - validation ensures that the value of this field is a proper serialized JSON list containing only string values by trying to deserialize the JSON string into a Java list of string objects.
- **titleLocArgs** - validation ensures that the value of this field is a proper serialized JSON list containing only string values by trying to deserialize the JSON string into a Java list of string objects.
- **maximum payload size** - according to GCM documentation, the maximum payload size of the "data" element cannot exceed 4KB. This validation step takes all notification elements into account that are rendered later under the GCM "data" element, and estimates the total payload size. To maintain backward compatibility with older mobile platform message formats, the maximum allowed payload size is actually only about one third of the 4KB. See Special Notification Elements, below.
If any validation step fails because an invalid value was sent, the returned HTTP status is 400. The push response provides detailed information, such as the push status, which is "3" denoting an input validation error; and the push status message, which indicates the element that has failed, and why.

Example response:

```
HTTP/1.1 400 Bad Request
X-SMP-LOG-CORRELATION-ID: b1922884-4612-47a5-8654-fc01792d0bf4
Content-Type: application/JSON

{  "status": {    "value": "GCM notification validation error: invalid time_to_live value: 2419201",    "code": 3  } }
```

These new validation steps also apply if the feature mentioned in JSON Payload Support for GCM Push Notifications is inactive.

**Send-to-Sync**

By introducing more granular validation functionality, the mobile platform GCM notification handler now supports GCM's "send-to-sync" functionality, which means the smallest possible payload can now be sent to GCM.

For more information see Cloud Messaging > GCM Connection Server > HTTP Connection Server, Send-to-sync: https://developers.google.com/cloud-messaging/http#send-to-sync.

Example of a send-to-sync request payload:

```
{}
```

The message sent to GCM is empty, except for the receiver's registration_id / GCM token, in a format similar to:

```
{ "to": "bk3RNwTe3H0:CI2k_HHwgIpoDKCI2vvDMExUdFQ3P1..." }
```

**Special Notification Elements**

This describes payload handling for several special notification elements, and provides examples.

**alert**

Since a GCM message does not contain an element that corresponds to the mobile platform "alert" element, the value supplied with this field is rendered under the GCM element "data". If the GCM JSON payload feature is inactive, the value is still rendered as the URL parameter "data.alert", but is transparent to the client. Keep in mind that "alert" does not equal the GCM message "title" element, which enables backward compatibility for clients who expect the alert to be part of the "data" payload.
The mobile platform notification element "badge" also has no corresponding GCM message element, and is rendered under the "data" element.

The mobile platform notification element "data" supports these content types:

- **String** - a normal string of characters
- **Serialized JSON** - a serialized JSON object

Previous versions of the mobile platform notification handler always sent the value as a string in the URL parameter "data.data". Now the notification handler tries to deserialize a JSON object structure from the parameter's content, and, if it succeeds, the JSON object structure renders under the GCM message element "data". To remain backward compatible, the value from "data" is also rendered under the GCM message element "data" using the key "data". This way the receiving client then can process the message as needed by accessing the deserialized JSON object, or accessing the serialized JSON string or plain text string. See the example below, which shows how the message is sent to GCM, and how the payload might be passed forward to the client.

The mobile platform notification element "customParameters" also allows you to specify values for "alert", "badge" and "data", which, if set, take priority over their corresponding notification element fields. Aside from this, the values are handled accordingly, and as described above, except the data element, which is additionally rendered under the GCM message element "data" and the key "cdata" as well. This approach provides backward compatibility, but reduces the maximum supported payload size, since information is duplicated.

**Example Requests**

The following examples show requests using the data and customParameters elements.

**data element**

Example request using the data element:

```
{
   "alert": "Update Available",
   "badge": 1,
   "data": 
   "{"\"version\":\"1.13\",\"size\":\"14MB\",\"details\":{}}"
}
```

The resulting GCM message payload:

```
{
   "to": "GCMTOKEN-1234567",
   "data": {
   "version": "1.13",
   "details": {
   "size": "14MB"
   },
   "data": "{"\"version\":\"1.13\",\"details\":{\"size\":\"14MB\"}}",
   "badge": "1",
   "alert": "Update Available"
   }
}
```

**customParameters element**
Example request using the customParameters element:

```
{
    "customParameters": {
        "gcm.alert": "Update Available",
        "gcm.badge": 1,
        "gcm.data": "{\"version\":\"1.13\",\"size\":\"14MB\",\"details\":{}}"
    }
}
```

The resulting GCM message payload:

```
{
    "to": "GCMTOKEN-1234567",
    "data": {
        "version": "1.13",
        "details": {
            "size": "14MB"
        },
        "data": "{\"version\":\"1.13\",\"details\":{"size":\"14MB\"}}",
        "cdata": "{\"version\":\"1.13\",\"details\":{"size":\"14MB\"}}",
        "badge": "1",
        "alert": "Update Available"
    }
}
```

The above examples show how "data" information is duplicated to remain backward compatible with older client applications, which expect data to be sent as string values under certain key names (such as "data", or "cdata"). The examples also show the proper GCM approach for sending a JSON payload, which is to deserialize the given "data" value into a JSON object structure.

---

### Resend Messages

If processing on the GCM side fails, you can automatically resend messages. Resend is scheduled when the GCM response contains an error code (currently only HTTP 400 error response codes), and includes the "Retry-After" HTTP header.

If these conditions occur, the original message persists in the database, and is picked up by the push dispatcher, which runs periodically. The previously extracted value from the "Retry-After" header is honored accordingly. Configure the maximum number of retries using the parameter "push.config.async_queue_max_retries". The default value is 3.

The current implementation sends only HTTP 400 error response codes. This does not comply with the Google developer’s documentation, which proposes using the "Retry-After" mechanism for 5<xxx> or 200+ error response codes. To comply, you must provision the 400 error code to handle the Resend mechanism for GCM Push API feature.

---

### Logging and Customer Events

SAP Cloud Platform mobile service for development and operations make an event log entry for every GCM error code returned from the GCM server. For details about the expected errors that can be returned by GCM,

Validation errors lead to a customer event log entry detailing the element for which validation failed, and why.

1.5.1.2.7.2.3  WNS Push Notification Payload Handling

WNS provides a native mechanism to send notifications to the back end. Back-end systems use the Push REST service to notify the mobile platform about notification messages it sends to devices.

Payload Handling for WNS

This section provides examples of JSON payload handling for WNS custom values.

**wnsNotification**

The wnsNotification entity provides payload handling for Windows. wnsNotification is validated with a higher priority than global configured properties (such as Data, Sound, Alert). The root element of the new entity, and all its direct subelements are optional.

**badge:**

The badge element represents a certain WNS push schema (wns/badge).

Element handling:

- Overrides the global badge configuration, whether it is a number or a string.
- The badge is sent as a separate request to the device.
- If a number, 0 clears the badge; values from 1-99 are shown as given; and any value greater than 99 are shown as 99+.
- If a string, the badge is shown as a predefined Windows glyph.


**rawData**

The rawData element represents a certain WNS push schema (wns/raw).

Element handling:

- Overrides the global data configuration.
- Any string can be configured as rawData, and sent to the device.

**version**

The version element sets the version property at the notification requests (tile, toast and badge).

Element handling:
- Any string configured as version is sent to the device.
- The default is ‘1.0’.

**lang**

The lang element sets the lang property for the notification requests (tile, and toast).

Element handling: defines the language property of the content.

**schema**

The schema element defines which notification schemas should be sent.

Element handling:
- A list containing up to four elements.
- Restricted to one of ‘BADGE’, ‘TILE’, ‘TOAST’ or ‘RAW’.
- The default sends all (in case content exists).

**tileTemplate**

The tileTemplate element sets the template property for tile schema at the binding attribute. Windows provides a selection of several predefined templates, which affect the final layout of the displayed notification at the device. The elements that can be selected depend on the device operating system version.

Element handling:
- Any string.
- Default value depends on the push content:
  - Text only: TileSquareText04
  - One picture and text: TileSquarePeekImageAndText04
  - Few pictures and text: TileWidePeekImageCollection04
  - One picture only: TileWideImage
  - Few pictures only: TileWideImageCollection


**toastTemplate**

The toastTemplate element sets the template property for toast schema at the binding attribute. Windows provides a selection of several predefined templates, which affect the final layout of the displayed notification at the device. The elements that can be selected depend on the device operating system version.

Element handling:
- Any string.
- Default value depends on the push content:
  - Text only: ToastText01
  - One picture and text: ToastImageAndText04
  - Few pictures and text: ToastImageAndText04
  - One picture only: ToastImageAndText04
  - Few pictures only: ToastImageAndText04

message
The message element defines the text attribute for tile and toast notifications.
Element handling:
- No, one, or more messages can be added.
- Any string.

baseUri
The baseUri element defines the baseUri property at the binding attribute for tile and toast. This property defines a central base URL which is used for all images.
Element handling: any string.

audio
The audio element defines the audio attribute for toast notifications.
Element handling: overrides the global sound configuration.
Properties:
- loop: [Boolean, optional] – defines whether the sound should be played in a loop.
- silent: [Boolean, optional] – defines whether the sound should be muted. If true, no sound is sent.
- src: [String, required] – defines the source of the sound file.

image
The image element defines the image attribute for tile and toast notifications.
Element handling: no, one, or more images can be added.
Properties:
- alt: [String, optional] – defines an alternative text if the image cannot be loaded.
- addImageQuery: [Boolean, optional] – allows Windows to add a query to the image src, the default is false.
- src: [String, required] – defines the source of the image.

commands
The commands element defines the commands attribute for toast notifications. Commands correspond to available actions that the user can take.
Element handling: no, one, or more commands can be added.
Properties: scenario [String,optional] – defines the scenario (alarm or incomingCall, the default is alarm).
Aggregations: command:
- id [String, optional] – specifies one command from the system-defined command list. (such as snooze, or dismiss).
- arguments [String, optional] – an argument string that can be passed to the associated app to provide specifics about the action that it should execute in response to the user action.
Example for JSON

The following example shows payload handling using JSON.

```json
{
    "wnsNotification": {
        "badge": "attention",
        "rawData": "WnsRawData",
        "version": "1.0",
        "baseUri": "http://foo.bar/base",
        "lang": "en-US",
        "schema": ["BADGE", "TILE", "TOAST", "RAW"],
        "tileTemplate": "TileWidePeekImageCollection04",
        "toastTemplate": "ToastImageAndText04",
        "message": ["message one", "message two", "message three"],
        "audio": {
            "loop": false,
            "silent": false,
            "src": "/foo/bar"
        },
        "image": [{
            "alt": "alternative text",
            "addImageQuery": true,
            "src": "/foo/de"
        }, {
            "alt": "alternative text2",
            "addImageQuery": false,
            "src": "/foo/de"
        }, {
            "src": "/foo/de"
        }],
        "commands": {
            "scenario": "alarm",
            "command": [{
                "id": "snooze",
                "arguments": "some arguments"
            }, {
                "id": "dismiss"
            }]
        }
    }
}
```

### 1.5.1.2.7.3 Notification Data Sent Through HTTP Headers

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

**Request**

**URL:** https://<HMC base URL>/Notification/<registration ID>
Note

- SAP recommends that you use the URLs provided in Notification Data Sent Using Push API.
- Applications built on platform and later should adopt the header format X-SMP-XXX. To maintain backward compatibility, applications built in earlier versions can continue to use the header format X-SUP-XXX. However, X-SUP-XXX headers will be removed in future releases.

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

Header format for notification data in platform and later:

```xml
<X-SMP-DATA>
```

APNS-specific headers
Use these APNS-specific HTTP headers to send APNS notifications via SAP Cloud Platform mobile service for development and operations:

Table 81:

<table>
<thead>
<tr>
<th>Header Structure (SAP Cloud Platform mobile service for development and operations 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. Click OK to dismiss 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>
<tr>
<td><code>&lt;X-SMP-APNS-ALERT-LAUNCH-IMAGE&gt;</code></td>
<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: ○ The previous snapshot ○ The image identified by the UILaunchImageFile key in the Info.plist file of the application ○ The Default.png.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-BADGE&gt;</code></td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-SOUND&gt;</code></td>
<td>Name of the sound file in the application bundle.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-DATA&gt;</code></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-specific headers**

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

<table>
<thead>
<tr>
<th>Header Structure (SAP Cloud Platform mobile service for development and operations and later)</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-APNS-CATEGORY&gt;</code></td>
<td>Identifies a category that is used to specify notification actions, such as contacts, or messages.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-APNS-CONTENT-AVAILABLE&gt;</code></td>
<td>Indicates whether the user can take action.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Header Structure (SAP Cloud Platform mobile service for development and operations and later)</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;X-SMP-GCM-COLLAPSEKEY&gt;</code></td>
<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>
<tr>
<td><code>&lt;X-SMP-GCM-DATA&gt;</code></td>
<td>Payload data, expressed as parameters prefixed with data and suffixed as the key.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-GCM-DELAYWHILEIDLE&gt;</code></td>
<td>(Optional) Represented as 1 or true for true, any other value for false, which is the default value.</td>
</tr>
<tr>
<td><code>&lt;X-SMP-GCM-TIMETOLIVE&gt;</code></td>
<td>Number of seconds that the message remains available on GCM storage if the device is offline.</td>
</tr>
</tbody>
</table>

### 1.5.1.2.7.4 SAP Gateway Notification Support

There are no specific handling requirements for sending notifications on the SAP Gateway side. mobile service for development and operations sends notifications using gateway-specific headers.

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

**Note**

Non-SAP Gateway back ends also use the headers listed below to send generic notifications; the back ends are unaware of the device platform.

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-sap-poke-title</td>
<td>Text of the alert message.</td>
</tr>
<tr>
<td>x-sap-poke-entriesofinterest</td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
</tbody>
</table>
### APNS
SAP Cloud Platform mobile service for development and operations converts the gateway notification headers into APNS notifications:

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-sap-poke-title</td>
<td>Text of the alert message.</td>
</tr>
<tr>
<td>x-sap-poke-entriesofinterest</td>
<td>Number that appears as the badge on the application icon.</td>
</tr>
<tr>
<td>x-sap-poke-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>
<tr>
<td>x-sap-poke-category</td>
<td>Identifies a category that is used to specify notification actions, such as contacts, or messages.</td>
</tr>
<tr>
<td>x-sap-poke-sound</td>
<td>Identifies the sound file to be played along with a push message when using the gateway headers.</td>
</tr>
<tr>
<td>x-sap-poke-content-available</td>
<td>Indicates that the user can take action.</td>
</tr>
</tbody>
</table>

### GCM
SAP Cloud Platform mobile service for development and operations converts the gateway notification headers into GCM notifications:

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-sap-poke-title</td>
<td>An arbitrary string (such as &quot;Updates Available&quot;) 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>x-sap-poke-data</td>
<td>Payload data. Size should not exceed 4KB.</td>
</tr>
</tbody>
</table>

### BIS/BES
SAP Cloud Platform mobile service for development and operations converts the gateway notification headers into BIS/BES notifications:

<table>
<thead>
<tr>
<th>Structure Header</th>
<th>Consists of</th>
</tr>
</thead>
<tbody>
<tr>
<td>x-sap-poke-data</td>
<td>BES/BIS notification data</td>
</tr>
</tbody>
</table>

### WNS
SAP Cloud Platform mobile service for development and operations converts the gateway notification headers into WNS notifications:

Table 87:

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

### 1.5.1.2.7.5 Notification Sent in URL Format

Notification data can also be sent by using the REST client, using URL arguments as part of the mobile platform push endpoint, or as the delivery address URL.

**Request**

**URL**: `http[s]://<host:port>/Notification/<application connection ID>?alert=<alert>&badge=<badge>&sound=<sound>&data=<data in text format>&category=<category_name>&content_available=true/false`

**URL**: `https://<HMC base URL>/Notification/<application connection ID>?alert=<alert>&badge=<badge>&sound=<sound>&data=<data in text format>&category=<category_name>&content_available=true/false`

All URL arguments (zero to many) are optional. The arguments are converted into device-type specific notifications as explained:

- **APNS**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<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>
<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>
<tr>
<td>category</td>
<td>Identifies a category that is used to specify notification actions, such as contacts, or messages.</td>
</tr>
<tr>
<td>Parameters</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>content_available</td>
<td>Indicates whether the user can take action.</td>
</tr>
</tbody>
</table>

- **GCM**

Table 89:

<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 90:

<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 91:

<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)**

Table 92:

<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 93:

<table>
<thead>
<tr>
<th>Header 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>
</tr>
</tbody>
</table>
### 1.5.1.2.8 Registering Clients for Native Push Notifications

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

#### Prerequisites

- Configure the registration ID.
- Configure the application to send push notifications.

- **Registering Android Clients** [page 279]
  
  Register and enable your Android device clients to receive push notifications.

- **Registering BlackBerry Clients** [page 280]
  
  Register and enable your BlackBerry device clients to receive push notifications.

- **Registering iOS Clients** [page 281]
  
  Register and enable your iOS device clients to receive push notifications.

- **Registering Windows 8 Desktop and Tablet Clients** [page 282]
  
  Register and enable your Windows 8 (desktop and tablet) devices to receive push notifications.

- **Registering Windows 8 Phone Clients** [page 283]
  
  Register and enable your Windows 8 Phone 8 client to receive push notifications.
1.5.1.2.8.1 Registering Android Clients

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

**Prerequisites**

- (Administrator) In the cockpit, configure the application for push notification by specifying the sender ID and API key.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

**Procedure**

1. If `AndroidGcmPushEnabled` is enabled, the sender ID is sent in the response. Upon successful client onboarding, the response indicates the GCM push is enabled.
2. If GCM is enabled and the sender ID is available, the client uses that sender ID to register itself with GCM and get its unique GCM registration ID.
3. Specify the POST method in the URL, and include the registration ID:

   ```html
   http://<host:port>/odata/applications/{<service version>}/{appid}/Connections/('{appcid}')
   Method : POST
   HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "MERGE"
   Body:
   <content type="application/xml">
     <m:properties>
       <d:AndroidGcmRegistrationId>{GCM registration ID}</d:AndroidGcmRegistrationId>
     </m:properties>
   </content>
   </entry>
   ```
1.5.1.2.8.2 Registering BlackBerry Clients

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

Prerequisites

- To configure push notifications for BIS, import a BIS certificate into the `smp_keystore.jks` and `keystore` files in the server configuration folder.
- (Administrator) In the cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

Enable push notifications in the application:

1. Update the application connection settings with the BES/BIS registration ID.
2. In your application, set the `BlackberryPushListenerPort` and `BlackberryDevicePin` properties.

```xml
http://<host:port>/odata/applications/{<service version>}/{appid}/Connections/('{appcid}')
Method : POST
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "MERGE"
Body: Http payload to update the blackberry (BES) device PIN and push port
<?xml version="1.0" encoding="utf-8"?>
  <m:properties>
    <d:BlackberryDevicePin> </d:BlackberryDevicePin>
    <d:BlackberryBESListenerPort><XXXX></d:BlackberryBESListenerPort>
  </m:properties>
</entry>

Body: Http payload to update the blackberry (BIS) device PIN and push port:
<?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>
```
1.5.1.2.8.3 Registering iOS Clients

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

Prerequisites

- (Administrator) In the cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.

Procedure

Enable push notifications in the application:

a. To receive the device token, implement the `application:didRegisterForRemoteNotificationsWithDeviceToken` method in your application delegate.

b. Update the `ApnsDeviceToken` and `DeviceType` properties via a POST request. The HTTP header must include the X-SMP-APPCID and Authorization headers.

```xml
https://<host:port>/odata/applications/{/service version}/appid/Connections/({'appcid'}
Method : POST
HTTP Headers : "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "MERGE"

Body:
<?xml version='1.0' encoding='utf-8'?>
xml:base="http://host:port/odata/applications/v1/com.example.IOS/">
  <id>https://{application URL}/odata/applications/{service version}/e2eTest/Connections('32552613-470f-45e0-8acc-b7d73d501682')</id>
  <content type="application/xml">
    <m:properties>
      <d:ApnsDeviceToken>{APNS device token received by the application from APNS}</d:ApnsDeviceToken>
      <d:DeviceType>iPhone</d:DeviceType>
    </m:properties>
  </content>
</entry>
```
1.5.1.2.8.4 Registering Windows 8 Desktop and Tablet Clients

Register and enable your Windows 8 (desktop and tablet) devices to receive push notifications.

Prerequisites

- (Administrator) In Mobile Service for Development and Operations cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

1. To obtain the channel URI, register the application with WNS. See Push notification overview (Windows Store apps) on the Windows Dev Center Web site.
2. Check the value that is returned from WnsPushEnable during registration, and if the value is true, continue with either the WNS or notification registration processing. Set the WnsChannelURI value that is received from the application.
3. Update the application connection settings with the registration ID:

```xml
https://host:port/odata/applications/{<service version>}/{appid}/Connections/('{appcid}')
Method : POST
HTTP Headers "Content-Type" = "application/atom+xml" and "X-HTTP-METHOD" = "MERGE"
Body:
  <content type="application/xml">
    <m:properties>
      <d:WnsChannelURI>{WNS Channel URI}</d:WnsChannelURI>
    </m:properties>
  </content>
</entry>
```
1.5.1.2.8.5 Registering Windows 8 Phone Clients

Register and enable your Windows 8 Phone 8 client to receive push notifications.

Prerequisites

- (Administrator) In Mobile Service for Development and Operations cockpit, configure the application for push notification.
- During application connection and registration, specify the device type.
- Include X-SMP-APPCID and Authorization headers in the HTTP header.

Procedure

1. In the cockpit, configure push notification. Specify the device type during application connection and registration, and ensure that the HTTP header includes the X-SMP-APPCID and Authorization headers.
2. To obtain the channel URI, register the application with the Microsoft Push Notification Service (MPNS). See Push notifications for Windows Phone on the Windows Phone Dev Center Web site.
3. Check the value of MpnsPushEnable that is returned during registration, and if the value is true, continue with either the MPNS or notification registration processing. Set the MpnsChannelURI value that is received from the application.
4. Using the ApplicationConnection ID (<appcid>) that is returned from the mobile platform registration call (in either the X-SMP-APPCID HTTP header or the ApplicationConnectionId property), update the MpnsChannelURI property for the application connection using the Channel URI returned by the application:

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

Get the service document for the application connection.

Usage

Retrieving the service document allows the client to discover the capabilities and locations of the available collections.

Request

URL: https://<HMC base URL>/odata/applications/<service version>/<appid>

HTTP Method: GET

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

Request Header Example

GET /odata/applications/v1/com.sap.myapp HTTP/1.1
Host: smpserver:8080
Authorization: Basic REVwMDAwMTpbml0aWFs

Response Body Example

```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>
```
### 1.5.1.2.10 Logging Out Users

Terminate an active user session with the user logout service. This is useful for multiple user scenarios.

The user logout service always returns HTTP status 204 (no content), even if there was no active session for the calling user at the point the call was made. The service does not require authentication, but it does require a session cookie header identifying the session to be terminated.

### Related Information

[Logout Service](#) [page 325]

### 1.5.1.2.11 Storage Service

As an application developer, you can use REST services to store device application configuration data in the server.

The application configuration data is stored based on user or device preference. The storage services stores flexible data structure and supports application-level, user-level, and device-level storage. SAP Cloud Platform mobile service for development and operations offers authorization and authentication schemes that secure the data.

### 1.5.1.2.12 Client Resources Service

As an application developer, you can use REST services to download resources from the server.

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Cloud Platform mobile service for development and operations Admin Cockpit.

---

### Table 95:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns service document</td>
</tr>
</tbody>
</table>
1.5.1.2.13 Client Log Upload Service

As an application developer, you can use REST services to upload client logs in the server. The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

1.5.1.2.14 Push as API Service

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices. Developers enable native push notification in the application code, and link the corresponding certificate with the mobile application at build time. Users download the application from a market place, such as Apple App Store, Google Play, or similar service, and, when a change occurs in the back end, a push notification is sent to mobile applications on devices that have push enabled.

1.5.1.2.15 Document Service

The Document Service API allows you to access Document Service repository through SAP Cloud Platform mobile service for development and operations. By using this API, you can consume the Document Service without performing the registration.

1.5.1.2.16 Client Usage Report Upload Service

The Client Usage Report Upload API allows the application to upload the client usage reports of SAP Cloud Platform mobile service for development and operations.

1.5.1.2.17 Role Service

As an application developer, you can use REST services to enable mobile devices to retrieve user name and roles associated with the user according to the System for Cross-domain Identity Management (SCIM) protocol. Role service allows you to get the logical roles for the current user, and you can use it to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

The service implements:

- The "/Me" Authenticated subject alias. This enables the client to use a URL in the form <base-URI>/ME as a URI alias for the user associated with the currently authenticated subject for a SCIM operation. The service provider may grant the request or redirect it.
- Multi-valued attributes. This enables the client to retrieve the roles associated with the user.
1.5.1.3 Reference

Describes REST API resources.

HTTP Headers and Cookies [page 288]
Use HTTP headers and cookies to retrieve application connection information.

HTTP Headers Used to Propagate User IDs [page 288]
Enables the back end to use the information in the X-SMP-ENDUSERNAME <username> header to identify the user who sent a request.

Supported Onboarding Services [page 289]
Lists the general naming conventions of the services supported for registration and onboarding purposes.

Proxy Responses [page 291]
Proxy responses include all the cookies and headers from the proxied back end.

Application Connections [page 291]
Methods for creating, updating, or reading application connections.

Error Codes and Message Formats [page 308]
The server returns different formats for error codes and messages according to different “Accept” values in request headers.

Authenticate Applications Using SAML 2.0 [page 308]
Initiate a REST service call to create SAML 2.0 assertion for authenticating the application security configuration.

Retrieve Customization Resource Bundles [page 315]
Download application resource bundles.

Accessing Services Through Proxy URLs [page 316]
To access a back end or Internet-based service, use a proxy URL that supports read, create, update, delete, merge and patch.

Feature Restriction Policies [page 318]
REST API methods for managing feature restriction policies for an application. You can get, update, or remove features enabled through the Java API, isEnabled(). Any enabled feature can be disabled by the administrator through the cockpit, providing additional control.

Upload Logs and Traces [page 321]
Upload client logs and Business Transaction XML (BTX) files for analysis.

Cross-Origin Resource Sharing Requests [page 324]
The cross-origin resource sharing (CORS) standard includes HTTP headers that allow you to control resource access from Web browsers. CORS also defines a preflight HTTP OPTIONS method that requests a list of the supported methods for a resource; if the requested method is supported, the actual HTTP request is sent.

Logout Service [page 325]
The user logout service provides functionality to terminate an active user session from the client.

Storage Service [page 326]
Storage service facilitates application developers to persist mobile application specific data, such as user preferences, user contextual data, and application configuration.

Client Resources Service [page 355]
The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Cloud Platform mobile service for development and operations Admin Cockpit.

**Client Log Upload Service [page 364]**

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

**Push as API Service [page 370]**

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

**Document Service [page 387]**

The Document Service API allows you to access Document Service repository through SAP Cloud Platform mobile service for development and operations. By using this API, you can consume the Document Service without performing the registration.

**Offline Store Upload Service [page 390]**

The Offline Store Upload API enables a client to upload offline store (database files) to the server. If Offline Store Upload is enabled for an application, you can upload the offline store (database files) into the server. The upload process creates a zip file that includes the uploaded database files, and is then saved into the server database.

**Client Usage Report Upload Service [page 397]**

The Client Usage Report Upload API allows the application to upload client usage reports for SAP Cloud Platform mobile service for development and operations.

**Role Service [page 403]**

This service allows you to get the logical roles that are assigned to the current user, which you can use to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

### 1.5.1.3.1 HTTP Headers and Cookies

Use HTTP headers and cookies to retrieve application connection information.

**Note**

In the current platform versions, applications should adopt the header format `X-SMP-XXX`. To maintain backward compatibility, applications built in earlier versions can continue to use the header format `X-SUP-XXX`. However, these headers will be removed from future releases, and you should update your applications to use the `X-SMP-XXX` header format.

### 1.5.1.3.2 HTTP Headers Used to Propagate User IDs

Enables the back end to use the information in the `X-SMP-ENDUSERNAME <username>` header to identify the user who sent a request.

Consider a scenario in which only the technical user ID, and not the user name of the mobile device appears in the back end. An administrator can now set the Propagate User Name option to true in SAP Cloud Platform...
mobile service for development and operations, which adds an X-SMP-ENDUSERNAME <username> HTTP header to outgoing requests, and securely passes the user name to the back end. The back-end service uses the information to apply additional filters that ensure that the information about the user who sent the request appears, even though the technical user made the request.

Related Information

Defining Connectivity [page 58]

1.5.1.3.3 Supported Onboarding Services

Lists the general naming conventions of the services supported for registration and onboarding purposes.

Whenever there is a change in the functionality newer version of onboarding services change in order to exchange data back and forth between client and server. Supported onboarding services to handle the requests which is OData compliant are:

<table>
<thead>
<tr>
<th>Onboarding Service Versions</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1</td>
<td>Initial version of the onboarding service. This service deviates from the standard OData compliant service in the sense that HTTP PUT requests can be used to change the individual properties, while parameters which are not included in the request remain same. When sending the complete entity payload, it however replaces the entity as expected. The service also includes a PATCH service, which must be accessed by tunneling the request as a POST request with the header X-HTTP-METHOD: MERGE. Example PATCH request*: POST/&lt;someentity&gt; X-HTTP-METHOD:MERGE {&quot;key&quot;:&quot;value&quot;} This request updates the key field of the entity.</td>
</tr>
</tbody>
</table>

Note

Furthermore, the feature vector field is modeled as a collection of complex types which is not a valid ODATA v2 construct. ODATA client libraries parsing the service metadata may report an error with v1 version of the service.
<table>
<thead>
<tr>
<th>Onboarding Service Versions</th>
<th>Impact</th>
</tr>
</thead>
</table>
| v2                          | Introduced changes in the semantics of the OData to ensure ODATA compliance. The PUT operation now updates/deletes any fields from the entity, if they are not included in the request payload. Clients must use a PATCH operation as described in the v1 service. Additionally, with v2 version, you can use the PATCH HTTP verb, instead of tunneling it with the special header in an HTTP POST request. Example POST request*:  

```plaintext
POST/<someentity>
X-HTTP-METHOD:MERGE{"key":"value"}
```

Or,

Sample PATCH request:

```plaintext
PATCH/<someentity>
X-HTTP-METHOD:MERGE{"key":"value"}
```

The feature vector in v2 is now modeled as an entity and referenced in this way from other entities. The metadata of the v2 service should be parsable by ODATA client libraries”. For more information, see Feature Restriction Policies. |
| v3                          | Introduced the device capabilities. For more information, see Create Application Connection with Capability Handling. |
| v4                          | Introduced a new onboarding contract for native applications only. This service has new application connection properties such as UserName, UserLocale, TimeZone, LastKnownLocation, CreatedAt, PushGroup, and Email. During onboarding, the GET operation can now receive a read-only property for UserName and CreatedAt. |
| latest                      | Always refers to the latest version of the onboarding service. **Note** The behavior of this service could change anytime if an updated version of the onboarding service is included in the server version. It is recommended to build your |
### Onboarding Service Versions

<table>
<thead>
<tr>
<th>Onboarding Service Versions</th>
<th>Impact</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>clients against a fixed server URL with a constant version.</td>
</tr>
</tbody>
</table>

#### Note

* Indicates a sample request and not all the headers are shown here.

### Related Information

Feature Restriction Policies [page 318]

### 1.5.1.3.4 Proxy Responses

Proxy responses include all the cookies and headers from the proxied back end.

### 1.5.1.3.5 Application Connections

Methods for creating, updating, or reading application connections.

#### Note

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

- **Metadata [page 292]**
  Get the metadata document, which includes the metadata for the application connection settings and proxy endpoints.

- **Retrieve Changed Settings and Connections Metadata [page 293]**
  You can retrieve only the changed settings and connections metadata.

- **Create Application Connection [page 294]**
  Create an application connection and initially set the application connection settings.

- **Create Application Connection with Capability Handling [page 296]**
  Enable the client to manage form factor and capabilities in the application connection.

- **Get Application Settings [page 302]**
  Retrieve application settings for the application connection.

- **Get Proxy Endpoints [page 303]**
  Get all proxy endpoints for the application connection.
Get Proxy Endpoint by Endpoint Name [page 304]
Get a specific endpoint by specifying the endpoint name.

Get Application Property Settings [page 305]
Get the specific property value for a property from the application settings.

Update Application Settings [page 306]
Update the application settings with the properties in the request.

Delete Application Connection [page 307]
Delete an application connection.

1.5.1.3.5.1 Metadata

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

Usage

Metadata documents are based on the OData standard and are required to implement application connection services.

Request

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/$metadata

**HTTP Method:** GET

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

**Request Header Example**

GET /odata/applications/v1/com.sap.myapp/$metadata HTTP/1.1  
Host: smpserver:8080  
Connection: Keep-Alive  
User-Agent: Apache-Http/4.1.3 (java 1.5)  
Authorization: Basic REVWMDBMTppbml0aWFs

**Request Body Example**

```xml
<?xml version="1.0" encoding="utf-8"?>
```
1.5.1.3.5.2 Retrieve Changed Settings and Connections Metadata

You can retrieve only the changed settings and connections metadata.

To retrieve changed application settings information, issue a GET request to this URL:

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/Connections({'<appcid>'})?If-None-Match="${ETag}"

**HTTP Method:** GET
Table 98:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;appid&gt;</code></td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td><code>&lt;service version&gt;</code></td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

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

### 1.5.1.3.5.3 Create Application Connection

Create an application connection and initially set the application connection settings.

#### Usage

All application connection settings are optional, the minimal body contains no properties at all. Mobile platform populates default values as needed.

#### Request

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/Connections

**HTTP Method:** POST

**Request Parameters**

Table 99:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;appid&gt;</code></td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
</tbody>
</table>

**Note**

If an application is configured for anonymous access in the cockpit, the registration is successful even if there are no credentials, or incorrect ones, in the authorization header.
**Parameter**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="">host:port</a></td>
<td>Mandatory</td>
<td>Host name should match the domain registered with mobile platform. If the requested domain name does not match, a default domain is used.</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

**Request Body Example**

```xml
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
  <title type="text"/>
  <updated>2014-06-15T02:23:29Z</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">18AA4813FB9E6393065BFEDADCD68173782A42599F3C9E2BF14F990F2D9F096</d:ApnsDeviceToken>
    </m:properties>
  </content>
</entry>
```

**Response**

**Table 100:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created New application connection settings are included in the response body.</td>
</tr>
</tbody>
</table>

**Related Information**

- Create Application Connection [page 238]
- Cross-Origin Resource Sharing Requests [page 324]
- Feature Restriction Policies [page 318]
1.5.1.3.5.4 Create Application Connection with Capability Handling

Enable the client to manage form factor and capabilities in the application connection.

The device sends its form factor (such as smartphone or tablet), and capabilities [such as purchaseOrder-display, or a wildcard (*) in case the device has all the capabilities] during registration, or when the application connection is updated. You can request a list of capabilities from the device. When the device user adds or removes a capability, the application connection is updated.

- **Create Capabilities upon Registration** [page 296]
  Store the device’s form factor and its capabilities when the device is registered.

- **Create Capabilities upon Update** [page 298]
  Update an existing application connection, using a POST method.

- **List Capabilities on a Device** [page 299]
  Obtain a list of all the capabilities of a device.

- **Delete Capabilities** [page 301]
  Delete an application capability, for example, a user may elect to remove the capability to receive an e-mail notification. The server is notified of this change to the application connection.

**Related Information**

- Push API Notification Scenarios [page 248]
- Push-to-Capability Scenario [page 258]
- Create Application Connection with Capability Handling [page 238]
- Feature Restriction Policies [page 318]

1.5.1.3.5.4.1 Create Capabilities upon Registration

Store the device’s form factor and its capabilities when the device is registered.

**Request**

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appID>/Connections

**HTTP Method** POST

**Request Parameters**
Table 101:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appID&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v3 onwards</td>
</tr>
</tbody>
</table>

Request Example

POST
http://localhost:8082/odata/applications/latest/
TESTAPP81de7af59f84f78a342ead3ad2a22a4/Connections
Content-Type: application/atom+xml Authorization: Basic
UDE5NDk3MDMyNDU60VJcmVOMT=\n<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
type="application/atom+xml;type=feed" title="Capability">
<content type="application/xml">
<m:properties>
<d:Category>push</d:Category>
<d:CapabilityName>purchaseOrder-display</d:CapabilityName>
</m:properties>
</content>
</entry>

Response Example

HTTP/1.1 201 Created
set-cookie: X-SMP-SESSID=14DBBA390DC7598A482B567006E109332628AB966D86E3867553E86092957BF9;
Path=/; HttpOnly
set-cookie: JTENANTSESSIONID_hmtenant1=IUXUlcqv8KCCwVYlL1TSUflsgqmyTgL13xLJnmEps%3D;
Path=/
set-cookie: X-SMP-APPCID=bd985307-497d-417e-b6fc-08f089d503f7; Expires=Wed,
11-Jul-2035 10:57:16 GMT; Path=/
set-cookie: X-SMP-APPCID=bd985307-497d-417e-b6fc-08f089d503f7; Expires=Wed,
11-Jul-2035 10:57:16 GMT; Path=/
set-cookie: data servic version: 1.0 date: Thu, 16 Jul 2015 10:57:16 GMT
location: http://localhost:8082/odata/applications/latest/
TESTAPP81de7af59f84f78a342ead3ad2a22a4/Connections('bd985307-497d-417e-
b6fc-08f089d503f7')
content-type: application/atom+xml; charset=utf-8 server: SAP
<entry
xmlns="http://localhost:8082/odata/applications/latest/
TESTAPP81de7af59f84f78a342ead3ad2a22a4/"

SAP Cloud Platform Mobile Service for Development and Operations
SAP Cloud Platform Mobile Service for Development and Operations
PUBLIC 297
Response

Table 102:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>The application connection has been created.</td>
</tr>
</tbody>
</table>

1.5.1.3.5.4.2 Create Capabilities upon Update

Update an existing application connection, using a POST method.

Usage

Update the form factor and capabilities when they are changed on the device (update app.connection).
Request

URL: https://<HMC base URL>/odata/applications/<service version>/<appId>/Connections(<registrationId>)

HTTP Method POST

Request Example

```
POST
HTTP Post with a header: X-HTTP-METHOD:MERGE
/odata/applications/latest/<appId>/Connections(<registrationId>)
```

Response

Table 103:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>The existing application connection has been updated.</td>
</tr>
</tbody>
</table>

1.5.1.3.5.4.3 List Capabilities on a Device

Obtain a list of all the capabilities of a device.

Request

URL: https://<HMC base URL>/odata/applications/<service version>/<appId>/Connections(<registrationId>)/Capability

HTTP Method GET

Request Parameters

Table 104:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appId&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;registrationId&gt;</td>
<td>Mandatory</td>
<td>The connection ID of the application instance that is interacting with the service</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v3 onwards</td>
</tr>
</tbody>
</table>

Request Example

```
GET
```
Response

Table 105:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>The list of capabilities has been created.</td>
</tr>
</tbody>
</table>
1.5.1.3.5.4.4 Delete Capabilities

Delete an application capability, for example, a user may elect to remove the capability to receive an e-mail notification. The server is notified of this change to the application connection.

Request

URL: https://<HMC base URL>/odata/applications/<service version>/<appId>/Capabilities(ApplicationConnectionId='<registrationID>',CapabilityName='<capability_name>',Category='<capability_category>')

HTTP Method DELETE

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appId&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>registrationId</td>
<td>Mandatory</td>
<td>The connection ID of the application instance that is interacting with the service</td>
</tr>
<tr>
<td>&lt;capability_name&gt;</td>
<td>Mandatory</td>
<td>The capability name used in the application, such as &quot;purchaseOrder-display&quot;.</td>
</tr>
<tr>
<td>&lt;capability_category&gt;</td>
<td>Mandatory</td>
<td>The capability category, such as &quot;push&quot;.</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v3 onwards</td>
</tr>
</tbody>
</table>

Request Example

DELETE http://localhost:8082/odata/applications/latest/TESTAPPea4a626ffccc4ef98dbf27da344326aa/Capabilities(ApplicationConnectionId='784b1fb8-4da5-4c87-a2b5-282bb4506253',CapabilityName='purchaseOrder-display',Category='push') Authorization: Basic UDE5NDA3MDMyNDU6U2VjcmV0MTI= X-SMP-APPCID: 784b1fb8-4da5-4c87-a2b5-282bb4506253

Response

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>The capability was removed from the application connection.</td>
</tr>
</tbody>
</table>
1.5.1.3.5.5 Get Application Settings

Retrieve application settings for the application connection.

Usage

You can retrieve application settings by either explicitly specifying the application connection ID, or by having the application connection ID determined from the call context (that is, from either the X-SMP-APPCID cookie or X-SMP-APPCID HTTP header, if specified). On the first call, you can simplify your client application code by having the application connection ID determined from the call context.

Request

**URL:** [https://HMC base URL]/odata/applications/<service version>/<appid>/Connections('<appcid>')

**HTTP Method:** GET

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;appcid&gt;</td>
<td>Mandatory</td>
<td>The connection ID of the application instance interacting with the service</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
<tr>
<td>&lt;user_name&gt;</td>
<td>Optional (read-only)</td>
<td>v4 onwards</td>
</tr>
</tbody>
</table>

**Request Header Example**

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

Table 109:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns service document</td>
</tr>
</tbody>
</table>

**Response Body Example**

```xml
<?xml version="1.0" encoding="UTF-8"?>
xmlns="http://www.w3.org/2005/Atom">
Connections<XXXX></id>
  <title type="text"/>
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">false</d:ApnsPushEnable>
      <d:ApnsDeviceToken m:null="true"/>
      <d:MpnsPushEnable m:type="Edm.Boolean">true</d:MpnsPushEnable>
      <d:MpnsPushEnable m:type="Edm.Boolean">true</d:MpnsPushEnable>
      <d:ProxyApplicationEndpoint>http://vmw3815.wdf.sap.corp:50009/sap/opu/sdata/
iwfnd/RMTSAMPLEFLIGHT/</d:ProxyApplicationEndpoint>
      <d:UploadLogs>false</d:UploadLogs>
      <d:WnsChannelURI m:null="true"/>
      <d:WnsPushEnable m:type="Edm.Boolean">false</d:WnsPushEnable>
    </m:properties>
  </content>
</entry>
```

**1.5.1.3.5.6 Get Proxy Endpoints**

Get all proxy endpoints for the application connection.

**Request**

**URL:** https://<HMC base URL>/odata/applications/<service version>/appid/Endpoints

**HTTP Method:** GET

**Request Parameters**
Table 110:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Endpoints HTTP/1.1
Host: smpserver:8080
X-SMP-APPCID=9dffe5e9-5768-47a6-8220-144a2e0c751d
```

Response

Table 111:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns service document</td>
</tr>
</tbody>
</table>

1.5.1.3.5.7 Get Proxy Endpoint by Endpoint Name

Get a specific endpoint by specifying the endpoint name.

Request

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/Endpoints('<endpoint>')

**HTTP Method:** GET

Request Parameters

Table 112:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;endpoint&gt;</td>
<td>Mandatory</td>
<td>The proxy endpoint name</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

Request Header Example

```
GET /odata/applications/v1/com.sap.myapp/Endpoints('endpoint1') HTTP/1.1
Host: smpserver:8080
X-SMP-APPCID=9dffe5e9-5768-47a6-8220-144a2e0c751d
```
Response

Table 113:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>404 - not found</td>
<td>Client tries to retrieve an endpoint that does not exist</td>
</tr>
<tr>
<td>400 - bad request</td>
<td>Client tries to fetch invalid property name</td>
</tr>
<tr>
<td>200 OK</td>
<td>OData response for endpoint-related information, which contains a remote URL and endpoint names and verifies whether anonymous access is allowed or not</td>
</tr>
</tbody>
</table>

1.5.1.3.5.8 Get Application Property Settings

Get the specific property value for a property from the application settings.

Request

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/Connections ('<registrationID>')/<property-name>

**HTTP Method:** GET

Request Parameters

Table 114:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;registrationID&gt;</td>
<td>Mandatory</td>
<td>The registration ID of the application instance that is interacting with the service</td>
</tr>
<tr>
<td>&lt;property-name&gt;</td>
<td>Mandatory</td>
<td>The property name can be appended to the URL to retrieve the value of a specific property</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

Request Header Example

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

Table 115:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns service document</td>
</tr>
</tbody>
</table>

1.5.1.3.5.9 Update Application Settings

Update the application settings with the properties in the request.

Table 116: Usage Information

<table>
<thead>
<tr>
<th>Condition</th>
<th>For Service Versions</th>
<th>Use HTTP Operation</th>
<th>Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>To replace an entity</td>
<td>v1 onwards</td>
<td>PUT</td>
<td>Add complete entity in the payload</td>
</tr>
<tr>
<td>To patch individual property</td>
<td>v1 onwards</td>
<td>POST</td>
<td>Add header X-HTTP-METHOD: MERGE and enter the properties to patch in the payload</td>
</tr>
<tr>
<td>within an entity</td>
<td>v2 onwards</td>
<td>PATCH</td>
<td>Add header X-HTTP-METHOD: MERGE and enter the properties to patch in the payload</td>
</tr>
</tbody>
</table>

Request

URL: https://<HMC base URL>/[public/]odata/applications/<service version>/<appid>/Connections('<registrationID>':<Version>)

HTTP Method: PUT

Request Parameters

Table 117:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>The application ID that uniquely identifies the application</td>
</tr>
<tr>
<td>&lt;registrationID&gt;</td>
<td>Mandatory</td>
<td>The registrationID of the application instance that is interacting with the service</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

Request Header Example

PUT /odata/applications/v1/com.sap.myapp/Connections('<XXXX>') HTTP/1.1
Response

Table 118:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>No response body</td>
</tr>
<tr>
<td>404</td>
<td>Not explicitly registered the client</td>
</tr>
</tbody>
</table>

1.5.1.3.5.10 Delete Application Connection

Delete an application connection.

Request

**URL:** https://<HMC base URL>/odata/applications/<service version>/<appid>/Connections('<appcid>')

**HTTP Method:** HTTP DELETE

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;appid&gt;</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>&lt;appcid&gt;</td>
<td>Mandatory</td>
<td>The connection ID of the application instance interacting with the service</td>
</tr>
<tr>
<td>&lt;service version&gt;</td>
<td>Mandatory</td>
<td>v1 onwards</td>
</tr>
</tbody>
</table>

**Request Header Example**

DELETE /odata/applications/v1/com.sap.myapp/Connections('b6d50e93-bca4-439d-9741-660a3cb56771') HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Host: smpserver:8080
Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.3 (java 1.5)
Authorization: Basic <XXXX>
Response

Table 120:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns service document</td>
</tr>
<tr>
<td>404</td>
<td>Explicitly not registered client</td>
</tr>
</tbody>
</table>

1.5.1.3.6 Error Codes and Message Formats

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

Table 121: Accept Header and Data Format

<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;title=&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;&lt;p&gt;&lt;b&gt;message string&lt;/b&gt; &lt;i&gt;error string&lt;/i&gt;&lt;/p&gt;&lt;p&gt;&lt;b&gt;description&lt;/b&gt; &lt;i&gt;error message&lt;/i&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;: {“lang”: &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.

1.5.1.3.7 Authenticate Applications Using SAML 2.0

Initiate a REST service call to create SAML 2.0 assertion for authenticating the application security configuration.

Usage

When an application initially connects to the server, a session is established. If the application is set up to be secured by SAML 2.0 authentication, the server responds with the header `com.sap.cloud.security.login:login-request` and SAML 2.0 authentication for the security configuration needs to take place in the application.
Note
This mechanism is also followed for any session that has not been authenticated, or has expired.

SAML 2.0 uses the HTTP redirect binding or HTTP POST to return the header response to the application. In this implemented, the server uses HTTP POST method to send the response.

![SAML authentication flow diagram](image)

**Request**

Issue an HTTP request to the server. If the server responds, the header indicates that SAML 2.0 authentication is required.

**URL**: http[s]://<HMC base URL>/SAMLAuthLauncher

**HTTP Method**: GET
Request Parameters
None

Request Body Example

1. When an application is initially launched, it sends a request that establishes a connection with the server. If the application is secured by SAML 2.0 authentication, the server sends a response containing these elements:
   ○ Response Header:
     ○ Name: com.sap.cloud.security.login
     ○ Value: login-request
     ○ Cookie X-SMP-SESSID
     ○ Status Code: HTTP-OK – 200
   Ensure that the response header contains the name and value `com.sap.cloud.security.login: login-request`, which indicates that SAML 2.0 authentication is required. If the response header is not returned, authentication does not take place.

   HTTP request header:
   ```
   Content-Type: application/atom+xml
   Accept: */*
   Accept-Encoding: gzip, deflate
   Accept-Language: en-US,en;q=0.8,hu;q=0.6
   Cookie: X-SMP-SESSID=72A4EEC7691ADBDA59E9413D4CFD2DCF57D3546AA9F62C118B25A600BFC43B0;
   BIGipServermobileciathanamobile.neo.ondemand.com=!
   xcgdDug8u1sOBsCBJH1n1K2lU+B0m3eQsQdVNZ
   IvSV6zbfwrS/o0K9sHCEpNgbUB16Zn//M99zCP\A=
   ```

   HTTP response headers:
   ```
   Access-Control-Allow-Origin: chrome-extension://
   hgmloofdfdffgphgcellddfbfbjejoo
   Access-Control-Allow-Credentials: true
   Set-Cookie: X-SMP-SESSID=D02CA63A6F411F52261C267F51610DB37A5A2C38EB38FF3CEDE757E9ED8B5A13;
   Path=/; Secure; HttpOnly
   P3P: CP="IDC DSP COR ADM DEVI TAIi PSA PSD IVAi IVD\i CONi HIS OUR IND
   CNT"
   com.sap.cloud.security.login: login-request
   Content-Type: text/html;charset=utf-8
   Transfer-Encoding: chunked
   Content-Encoding: gzip
   Vary: Accept-Encoding
   Date: Mon, 19 Jan 2015 08:18:57 GMT
   Server: SAP
   ```

   Request body:
   ```xml
   <?xml version='1.0' encoding='utf-8'?>
   <entry xmlns="http://www.w3.org/2005/Atom"
   <title type="text"></title>
   <updated>2012-06-15T02:23:29Z</updated>
   <author>
   <name/>
   </author>
   <category term="applications.Connection" scheme="http://
   schemas.microsoft.com/ado/
   2007/08/dataservices/schema"/>
   <content type="application/xml">
   <m:properties>
```
2. When the response is received, the application starts the authentication process, using the web view. The web view must use the X-SMP-SESSID cookie to start authentication for the security configuration.

```javascript
/* Now that you have received com.sap.cloud.security.login: login-request response header and SAML2 JavaScript redirect in the response body.

You need to start SAML2 authorization to obtain SAML2 related cookies. Note that the X-SMP-SESSID cookie received from the first response is carried over to the authorization request.
This request should be executed in a web view/web browser in order for the JavaScript redirect to be executed. */
Issue a GET method on the request URL:
GET https://mobileciathanamobile-x054703e3.neo.ondemand.com/SAMLAuthLauncher
```

Request headers:

```
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding:gzip, deflate, sdch
Accept-Language:en-US,en;q=0.8,hu;q=0.6
Cache-Control:max-age=0
Connection:keep-alive
Cookie:oucrszoqvbmoviudlsofnanai=Had%2BMvTnHp9VrB%2B9siILH1hzGmPfNyvv%2FBwOIvFEE1W0BEC0%2BNk14eYparug6I71WnQxbObzb5f1yWS88BcXoA9r%2FX2XH2%2FH5%2FkcmNtbo6Uy%2B2F5sVyHMj;
BIGipServermobileciathanamobile.neo.ondemand.com=!xgcdDpuqEv1sOsCB/JHa1kKZ1u+80m3eQ5dVzIvSV6zbwvrs/oo99sHCEpNgUB162n/M99zCFTA=; oucrsrgqnyggsdmvesujgwydp=Zb08ta%FNOPNibwJiQHo5j1cc7nzA7j6XypjoPegzrFwiNsrpV7%2F; 
X-SMP-SESSID=7EDE2B43E2310927CE199F3086FCAC481AC74CC165281E585BD2B3B33ACEB60;
JTENANTSESSIONID_x054703e3=31B8wXsrrtK8h4%2F9ckW3n72WEmsG8RbSg0j1m0zjB%3D
Host:mobileciathanamobile-x054703e3.neo.ondemand.com
Referer:https://accounts.sap.com/saml2/idp/sso/accounts.sap.com
User-Agent:Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36
```

Response headers:

```
Connection:Keep-Alive
Content-Encoding:gzip
Content-Length:166
Date:Mon, 19 Jan 2015 08:32:30 GMT
Location:https://mobileciathanamobile-x054703e3.neo.ondemand.com/SAMLAuthLauncher?finishEndpointParam=someUnusedValue
Server:SAP
Vary:Accept-Encoding
```

Request Payload: N/A

Response: N/A

3. To complete SAML 2.0 authentication, the following operation takes place automatically:
   1. The web view is redirected to the SAML 2.0 identity provider sign-on login URL.
2. After successful login, the web view is redirected to the SAML 2.0 assertion to check the response from the identity provider at:

<host:port>/saml/sso

3. The SAML assertion checks the response and creates an authenticated session for the application. The web view is redirected to:

<host:port>/SAMLAuthLauncher?finishEndpointParam=someUnusedValue

/*After successful authentication on the IDP, you are redirected to the SAMLAuthLauncher endpoint of the SMP server. */ Issue a POST method on the request URL:

POST https://mobileciathanamobile-x054703e3.neo.ondemand.com/SAMLAuthLauncher

Request header:

Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Encoding:gzip, deflate
Accept-Language:en-US,en;q=0.8,hu;q=0.6
Cache-Control:max-age=0
Connection:keep-alive
Content-Length:6332
Content-Type:application/x-www-form-urlencoded
Cookie:oucrszoqvbmoviudlsofnanai=Had%2BMvTnHp9VrB%2BsiilHHzGmPfNyv
%2F1bW
Host:mobileciathanamobile-x054703e3.neo.ondemand.com
Origin:https://accounts.sap.com
Referer:https://accounts.sap.com/saml2/idp/sso/accounts.sap.com
User-Agent:Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36

Response headers:

Connection:Keep-Alive
Content-Encoding:gzip
Content-Length:141
Date:Mon, 19 Jan 2015 08:32:30 GMT
Location:https://mobileciathanamobile-x054703e3.neo.ondemand.com/ SAMLAuthLauncher
P3P:CP="IDC DSP COR ADM DEVI TAIi PSA PSD IVAI IVDi CONi HIS OUR IND CNT"
Server:SAP
Set-Cookie:X-SMP-SESSID=7EDE2B43E2310927CE199F3086FCAC4E81AC74CC165281E558BD2B3B33ACEB60;
Path=/; Secure; HttpOnly
Set-Cookie:oucrsszczkurrnvejldzyiel_anchor=0; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:10 GMT
Set-Cookie:oucrsszczkurrnvejldzyiel=0; Max-Age=0; Expires=Thu, 01-Jan-1970 00:00:10 GMT; Domain=.ondemand.com; Path=/; Secure; HttpOnly
Set-Cookie:JSESSIONID_x054703e3=01BnwXsxxrtk8F4u%2F9ckW3n7ZWKemG8hBsg0J1m0zjdz%3D; Domain=.ondemand.com; Path=/; Secure; HttpOnly
Vary:Accept-Encoding

Request payload:

SAMLResponse:XXX
RelayState:oucrsszczkurrnvejldzyiel
Status 302 Found

Response: N/A
4. After the web view is redirected, close the view, then invoke the original REST service call by using the authenticated session (cookie) from the web view.

Request:

```java
/* Lastly, the server executes a redirect to the same base url and adds query params to signal the clients that the SAML2 flow has finished successfully. */
GET https://mobileciathanamobile-x054703e3.neo.ondemand.com/SAMLAuthLauncher?
finishEndpointParam=someUnusedValue
```

Request headers:

- Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp, */ ; q=0.8
- Accept-Encoding: gzip, deflate, sdch
- Accept-Language: en-US, en; q=0.8, hu; q=0.6
- Cache-Control: max-age=0
- Connection: keep-alive
- Cookie: oucrszqybmoviulsofnanai=Had%2BMvTnHp9VrB%2B9siILHzGmPfNyvv%2FIVFEE1W4OB
- Host: mobileciathanamobile-x054703e3.neo.ondemand.com
- User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36

Response headers:

- Content-Length: 0
- Date: Mon, 19 Jan 2015 08:32:30 GMT
- Server: SAP

Response: N/A

Re-send the registration request.

Request:

```http
```

Request Payload:

```xml
<?xml version='1.0' encoding='utf-8'?>
<entry xmlns="http://www.w3.org/2005/Atom"
<title type="text"/>
<updated>2012-06-15T02:23:29Z</updated>
<author>
<name/>
</author>
<content type="application/xml">
<m:properties>
<d:DeviceType>iPad</d:DeviceType>
<d:DeviceModel m:isNull="true"/>
</m:properties>
</content>
</entry>
```

Status 201 Created
**Note**
At any point when the SAML2 session is invalid, or the binding cookies on the client side expire, you must encounter SAML2 form response.

**1.5.1.3.8 Retrieve Customization Resource Bundles**

Download application resource bundles.

**Note**
Application developers can customize and retrieve resource bundles. If the values of `<resourceBundlename>` and `<resourceBundleVersion>` are specified in the URL, the resource bundle is returned in the response body as a stream; otherwise, the resource bundle that is bound to the application is returned. The resource-bundle extension is in the response header X-BUNDLE-EXTENSION.

If the resource bundle is not found in mobile platform, error code 404 is returned. You cannot issue other HTTP methods (PUT/POST/DELETE) at the above URL.
Request

**URL:** https://<HMC base URL>/bundles/<appid>/[<resourceBundleName>:<Version>]

**HTTP Method:** GET

**Request Parameters**

Table 122:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appid</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application</td>
</tr>
<tr>
<td>resourceBundleName</td>
<td>Optional</td>
<td>Returns the resource bundle</td>
</tr>
<tr>
<td>Version</td>
<td>Optional</td>
<td>Returns version of the resource bundle</td>
</tr>
</tbody>
</table>

**Request Body Example**

GET /bundles/com.sap.myapp/MyApp:1.0 HTTP/1.1
Cookie: X-SMP-APPCID=<XXXX>; X-SMP-SESSID=<XXXX>
Host: smpserver:8080
Authorization: Basic <XXXX>

Response

Table 123:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 OK</td>
<td>Returns resource bundle content</td>
</tr>
<tr>
<td>404 Not Found</td>
<td>Resource bundle is not found</td>
</tr>
</tbody>
</table>

**1.5.1.3.9 Accessing Services Through Proxy URLs**

To access a back end or Internet-based service, use a proxy URL that supports read, create, update, delete, merge and patch.

**Note**

Verify that all the URLs to be proxied are whitelisted.
Usage

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

HTTP Operations

Table 124:

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Request URL</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>https://&lt;HMC base URL&gt;/[public]/{connectionName}/[&lt;Collection&gt;]</td>
<td>Retrieve data from the back end through the mobile platform.</td>
</tr>
<tr>
<td>POST</td>
<td>https://&lt;HMC base URL&gt;/[public]/{connectionName}/[&lt;Collection&gt;]</td>
<td>Requests the server to accept the data in the request message body.</td>
</tr>
<tr>
<td>PUT</td>
<td>https://&lt;HMC base URL&gt;/[public]/{connectionName}/[Collection]/('&lt;EntryID&gt;')</td>
<td>Update an entry in the back end.</td>
</tr>
<tr>
<td>DELETE</td>
<td>https://&lt;HMC base URL&gt;/ {connectionName}/[Collection]/('&lt;EntryID&gt;')</td>
<td>Delete an entry from the back end.</td>
</tr>
<tr>
<td>MERGE</td>
<td>https://&lt;HMC base URL&gt;/[public]/{connectionName}/[Collection]/('&lt;EntryID&gt;')</td>
<td>Incrementally updates without replacing all the content of an entry.</td>
</tr>
<tr>
<td>PATCH</td>
<td>https://&lt;HMC base URL&gt;/[public]/{connectionName}/[Collection]/('&lt;EntryID&gt;')</td>
<td>Performs partial updates without replacing all the content of an entry. A PATCH request updates only the properties indicated in the request body.</td>
</tr>
</tbody>
</table>

The pattern of the URL path depends on the rewrite mode configured for the backend connection. For information on rewrite modes, see Creating a Back-End Connection [page 135].
If an application is configured for anonymous access, the request-response is made using the same user credentials provided in the "Allow anonymous connections" field for defining the back-end connection in the cockpit.

**Request Header Example**

```
X-SMP-APPCID : <Application connection Id received in the response of the onboarding xml>
Content-Type : application/atom+xml
X-Requested-With : XMLHttpRequest
Authorisation : <Base 64 encoded value of Authorization>
```

**Response**

Table 125:

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>200</td>
<td>Returns data from back end</td>
</tr>
<tr>
<td>POST</td>
<td>201</td>
<td>Returns when server accepts the data</td>
</tr>
<tr>
<td>PUT</td>
<td>204</td>
<td>Returns on successful update of entry in the back end</td>
</tr>
<tr>
<td>DELETE</td>
<td>204</td>
<td>Returns on successful deletion of entry in the back end</td>
</tr>
<tr>
<td>MERGE</td>
<td>204</td>
<td>Returns on successful merge of entry in the back end</td>
</tr>
<tr>
<td>PATCH</td>
<td>204</td>
<td>Returns on successful merge of entry in the back end</td>
</tr>
</tbody>
</table>

**Note**
No information is returned from a DELETE request.

**Related Information**

Creating a Destination [page 135]

### 1.5.1.3.10 Feature Restriction Policies

REST API methods for managing feature restriction policies for an application. You can get, update, or remove features enabled through the Java API, `isEnabled()`. Any enabled feature can be disabled by the administrator through the cockpit, providing additional control.

Get Feature Restriction Policy [page 319]
Get the feature restriction (or vector) policy for an application.
Update Feature Restriction Policy [page 320]
Update the feature restriction (or vector) policy for an application.

Remove Feature Restriction Policy [page 321]
Remove a feature (or vector) restriction policy from an application.

Related Information

Supported Onboarding Services [page 289]
Create Application Connection with Capability Handling [page 296]
Create Application Connection [page 294]

1.5.1.3.10.1 Get Feature Restriction Policy

Get the feature restriction (or vector) policy for an application.

Request

URL: http[s]://<HMC base URL>/Admin/FeatureVectorPolicy/<appid>

HTTP Method: GET

Request Parameters

Table 126:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appid</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application.</td>
</tr>
</tbody>
</table>

Request Header Example

Content-Type: application/json
Authorization: Basic <admin credentials>

Request Body Example

GET https://<host:port>/Admin/FeatureVectorPolicy/<appid>/

Response

```json
{
  "applicationID": "<appid>",
  "appVersion": "1.0",
  "name": "Barcode",
  "displayName": "Barcode Scanner",
  "id": ...
}
```
1.5.1.3.10.2 Update Feature Restriction Policy

Update the feature restriction (or vector) policy for an application.

Request

**URL:** http[s]://<HMC base URL>/Admin/FeatureVectorPolicy/<appid>

**HTTP Method:** PUT

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appid</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an applica-tion.</td>
</tr>
</tbody>
</table>

**Request Header Example**

Content-Type: application/json
Authorization: Basic <admin credentials>

Response

**Response example:**

```json
[{
  "applicationID": "<appid>",
  "appVersion": "1.0",
  "name": "accelerometer",
  "displayName": "accelerometer",
  "id": "org.apache.cordova.accelerometer",
  "version": "3.0",
  "description": "Plugin for accelerometer",
  "jsModule": "navigator.accelerometer",
  "whitelist": "*"
}]
```
1.5.1.3.10.3 Remove Feature Restriction Policy

Remove a feature (or vector) restriction policy from an application.

Request

URL: [https://<HMC base URL>/Admin/FeatureVectorPolicy/<appid>](https://<HMC base URL>/Admin/FeatureVectorPolicy/<appid>)

HTTP Method: DELETE

Request Parameters

Table 128:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>appid</td>
<td>Mandatory</td>
<td>ID that uniquely identifies an application.</td>
</tr>
</tbody>
</table>

Request Header Example

```
Content-Type: application/json
Authorization: Basic <admin credentials>
```

Request Body Example

```
DELETE https://<host:port>/Admin/FeatureVectorPolicy/
```

Response

```
[{
  "applicationID": "<appid>",
  "appVersion": "1.0",
  "name": "accelerometer",
  "displayName": "accelerometer",
  "id": "org.apache.cordova.accelerometer",
  "version": "3.0",
  "description": "Plugin for accelerometer",
  "jsModule": "navigator.accelerometer",
  "whitelist": "*"
}]
```

1.5.1.3.11 Upload Logs and Traces

Upload client logs and Business Transaction XML (BTX) files for analysis.

Usage

SAP Cloud Platform mobile service for development and operations provides a generic REST service for uploading client log files, BTX, and other trace files to the database.
In the cockpit, the administrator enables log upload settings for application connections to view the logs and traces from the cockpit.

Upload Client Logs [page 322]
Upload Client Logs API allows the application to upload client logs to the database.

1.5.1.3.11.1 Upload Client Logs

Upload Client Logs API allows the application to upload client logs to the database.
Upload Client Logs service supports only one Runtime API using the POST operation.

1.5.1.3.11.1.1 Runtime APIs

The Upload Client Logs API allows the application to upload the client logs to the server for further analysis.

Table 129: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>POST Client Log Upload Service [page 365]</td>
</tr>
</tbody>
</table>

1.5.1.3.11.1.1.1 POST Upload Client Logs

Upload Client Logs API allows the application to upload the client logs to the server for further analysis. This method allows you to upload client logs for an application. If the client log upload option is enabled for the specified application, then the client logs get uploaded.

HTTP Method: POST.

URL: https://<HMC base URL>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}

Request Parameters

Table 130:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application identifier</td>
<td>String</td>
</tr>
<tr>
<td>deviceId</td>
<td>Optional</td>
<td>Device identifier. Its value will be saved with uploaded log if specified.</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>-------------</td>
<td>--------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>clientlogs</td>
<td>Required</td>
<td>Client logs. It can be zipped data, multipart data or simple binary data. Log format: <code>#Date time#Severity#CorrelationId#Source#Location#Message#</code> The format of Date time is &quot;YYYY-MM-DDThh:mm:ss.sTZD&quot;, such as &quot;2014-07-01T17:16:08.637+02:00&quot;. The valid values of Severity are DEBUG, INFO, WARN, ERROR and FATAL. The CorrelationId value is optional, and can be left as empty if client cannot provide it for each log message. The Message value is log message generated by client, and # character in it must be escaped to avoid corrupt uploading.</td>
<td></td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 131:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Client logs uploaded.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid log format.</td>
</tr>
<tr>
<td>403</td>
<td>Client log upload is not enabled for the application.</td>
</tr>
<tr>
<td>404</td>
<td>No application found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
1.5.1.3.12 Cross-Origin Resource Sharing Requests

The cross-origin resource sharing (CORS) standard includes HTTP headers that allow you to control resource access from Web browsers. CORS also defines a preflight HTTP OPTIONS method that requests a list of the supported methods for a resource; if the requested method is supported, the actual HTTP request is sent.

CORS Headers

To enable CORS requests, configure the header parameters. See Configuring Cross-Origin Resource Sharing.

Table 132: CORS Configuration Properties

<table>
<thead>
<tr>
<th>Header</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Access-Control-Allow-Origin</td>
<td>A comma-delimited list of URIs that can access the resource, for example, <a href="http://example.com">http://example.com</a> or http://*.example.com.</td>
</tr>
<tr>
<td>Access-Control-Expose-Headers</td>
<td>A comma-delimited list of response headers that browsers can access.</td>
</tr>
<tr>
<td>Access-Control-Max-Age</td>
<td>The number of seconds for which the results of a request can be cached.</td>
</tr>
<tr>
<td>Access-Control-Allow-Credentials</td>
<td>Always set to On. The server includes cookies when it submits requests.</td>
</tr>
<tr>
<td>Access-Control-Allow-Methods</td>
<td>A comma-delimited list of HTTP methods (such as GET and DELETE) that are allowed when accessing the resource.</td>
</tr>
<tr>
<td>Access-Control-Allow-Headers</td>
<td>A comma-delimited list of HTTP request headers that you can specify in requests. The server always includes the following headers in its responses: accept, authorization, maxdataserviceversion, and x-smp-appcid.</td>
</tr>
</tbody>
</table>

Web Browser Restrictions

The browser restrictions are:

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

Programmable HTTP Requests

Developers can send HTTP requests in AJAX, in which case, JavaScript sends the preflight HTTP OPTIONS request. The following is a jQuery AJAX example:

```javascript
/*
Create Application Registration/Connection
*/
$.ajax{
```
The user logout service provides functionality to terminate an active user session from the client.

The service always returns HTTP status 204 (no content), even if there was no active session for the calling user at the point the call was made, and does not require authentication (but it does require a session cookie header identifying the session to be terminated). Implement the functionality using the logout service API.

Two options are available - POST and DELETE. The POST method does the same thing as DELETE, but is the option to use if the configuration includes a firewall.

**POST User Logout Request [page 326]**

Terminate an active user session from the client using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

**DELETE User Logout Request [page 326]**

Terminate an active user session from the client using the DELETE method (if the configuration includes a firewall, use the POST method instead).
1.5.1.3.13.1 POST User Logout Request

Terminate an active user session from the client using the POST method. Use the POST method if the configuration includes a firewall (otherwise use DELETE).

Request

URL: http[s]://<HMC base URL>/mobileservices/session/logout

HTTP Method POST

Request Body Example

```plaintext
POST https://<host:port>/mobileservices/session/logout HTTP/1.1
Accept-Encoding: gzip, deflate
```

1.5.1.3.13.2 DELETE User Logout Request

Terminate an active user session from the client using the DELETE method (if the configuration includes a firewall, use the POST method instead).

Request

URL: http[s]://<HMC base URL>/mobileservices/session

HTTP Method DELETE

Request Body Example

```plaintext
DELETE https://<host:port>/mobileservices/session HTTP/1.1
Accept-Encoding: gzip, deflate
Host: <host>
    Connection: Keep-Alive
User-Agent: Apache-HttpClient/4.1.1 (java 1.5)
```

1.5.1.3.14 Storage Service

Storage service facilitates application developers to persist mobile application specific data, such as user preferences, user contextual data, and application configuration.

Storage service does not contain business-specific data. Although you can create, update, or delete all the storage configurations using Runtime API and Admin API, there are some restrictions or conditions while using Runtime API.
Using Storage service, data can be stored at three levels:

- Application storage: persists global configuration and application default settings.
- User storage: persists user preferences
- Device storage: persists installation-specific or device-specific settings.

**Application Storage**

If you create an application storage configuration using Runtime API, then it is accessible to all users. If you get the user storage or device storage for the application with `overwrite=true` query parameter, then the application level configuration gets merged into the response.

While creating or updating application storage configuration using Admin API, a `__metadata` property (com.sap.mobile.server.storage.admin.v1.StorageMetadata type) can be included for the application configuration to define read and write right control of the runtime API. If no ReadRoles or WriteRoles are defined for an application configuration, then the application configuration can be accessed by all users, similar to a Runtime API.

Once the ReadRoles are defined for an application configuration, then the users who have the required roles can read the application configuration using Runtime API. Similarly, only these users can get the merged application configuration, while getting the user storage or device storage configuration with `overwrite=true` query parameter using the Runtime API. Once the WriteRoles are defined for an application configuration, then the users who have the required roles can modify the application configuration using Runtime API.

The required roles are mandatory for reading or writing the application configuration using Runtime API, if an application storage configuration is created by Runtime API, but modified by Admin API to add ReadRoles or WriteRoles.

**User Storage**

Users can only access the user configurations that belong to them by using the Runtime API. When you get a user storage configuration with `overwrite=true` query parameter using the Runtime API, the response includes the merged application configuration, which only the users who have the required ReadRole can access it.

If a property is defined on both application storage and user storage, the one in user storage gets included in response while getting with `overwrite=true` query parameter. If no one is defined in the user storage, then the one on the application storage gets included in response while getting with `overwrite=true` query parameter.

**Device Storage**

User can only access the device configuration that belongs to them. While getting the device storage configuration for an application with `overwrite=true` query parameter using the Runtime API, the response includes the merged user storage configuration and application storage configuration, which only the users who have the required ReadRole can access it. The device storage property has the highest priority, and the application storage property has the lowest priority.

**Samples for Starge Service**

**Example**

In most mobile applications, the configuration used in the Application Storage is used as the default settings, and the end users can customize the settings in the User Storage or Device Storage level in order...
to provide personalized settings for the user or device. In such cases in order to make sure that only the Administrator can modify the Application Storage configuration and other users can only view it (by Application Storage Runtime API or user/device Storage Runtime API with "overwrite=true" query parameter), the application storage configuration can be configured with empty ReadRoles and ["Administrator"] WriteRoles.

**Example**

The application storage configuration "HomeScreen" is created for application "mobileApp".

**Sample Code**

```
{
  "ApplicationId": "mobileApp",
  "HomeScreen": {
    "Menu": {
      "AutoHiding": false,
      "Style": "blue",
      "AutoHiding@odata.type": "Edm.Boolean",
      "Style@odata.type": "Edm.String"
    },
    "ShortCuts": ["Overview", "Profile"],
    "__metadata": {
      "ReadRoles": [],
      "WriteRoles": ["Administrator"]
    },
    "Menu@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
    "ShortCuts@odata.type": "Collection(Edm.String)",
    "__metadata@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageMetadata"
  },
  "HomeScreen@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject"
}
```

Only users with the Administrator role can modify these settings using Runtime API, but all other users can get these settings.

Then, the mobile application can get the settings on the device using `https://<HMC base URL>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/HomeScreen?overwrite=true` and fill the screen with values you received. The response body looks like:

**Output Code**

```
"Menu": {
  "AutoHiding": false,
  "Style": "blue"
},
"ShortCuts": ["Overview", "Profile"]
```

Assume the user configures the Menu/AutoHiding property value to true at device level:

PATCH `https://<HMC base URL>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/HomeScreen` body:

```json
{
  "Menu": {
    "AutoHiding": true,
    "Style": "blue"
  },
  "ShortCuts": ["Overview", "Profile"]
}
```

If you want to provide application level settings only to particular users and not all users (or settings provisioned in mobile app), then the ReadRoles can be defined, so that only users who have the Read role can get these settings.

For example, you want to provide a 2-factor authentication for managers to post some forms. Then another configuration "2factorAuth" can be defined for "mobileApp" application with `specify ReadRoles to "Manager"`. Then, the whole application storage for "mobileApp" looks like:

```json
{
  "ApplicationId": "mobileApp",
  "HomeScreen": {
    "Menu": {
      "AutoHiding": false,
      "Style": "blue",
      "AutoHiding@odata.type":"Edm.Boolean",
      "Style@odata.type":"Edm.String"
    },
    "ShortCuts": ["Overview", "Profile"],
    "__metadata": {
      "ReadRoles": [],
      "WriteRoles": ["Administrator"]
    },
    "Menu@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageObject",
    "ShortCuts@odata.type": "Collection(Edm.String)",
    "__metadata@odata.type": "#com.sap.mobile.server.storage.admin.v1.StorageMetadata"
  },
  "2factorAuth": {
    "RequestToken": {
      "url": "https://2factor.server/requesttoken/phone",
      "method": "POST",
      "parameters": ["username", "phone"]
    },
    "VerifyToken": {
      "url": "https://2factor.server/verify",
      "method": "POST",
      "parameters": ["token"]
    }
  }
}
```
Then, the `https://<HMC base URL>/mobileservices/application/mobileApp/Storage/v1/runtime/application/mobileApp/device/mydevice/2factorAuth?overwrite=true` can return settings according to the login user’s role, and mobile app can show screens and control flows according to the 2factorAuth settings.

The Application Storage configuration can also be used to persist settings for server application, if both ReadRoles and WriteRoles are defined as a role (or roles) which never belong to any users. Then, your server application can read/write these application storage configuration using a technical user who has the role.

Storage service supports two type of APIs:

- Admin APIs
- Runtime APIs

**Related Information**

Runtime APIs [page 330]
Admin APIs [page 345]

**1.5.1.3.14.1 Runtime APIs**

Used by mobile application to access settings at all the three storages level during runtime.

The runtime API can be accessed by both application authentication and origin authentication URL patterns. Currently, the runtime APIs only addressed origin authentication URL pattern. For example,

- Origin authentication URL pattern: `https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentUser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>/?overwrite=<overwrite>`
- Application authentication URL pattern: `https://<HMC base URL>/mobileservices/application/<applicationId>/<serviceName>/. Request is authenticated by application’s security configuration (configured in Mobile Service for Development and Operations cockpit).

If mobile application only needs to access default settings, it can directly use application storage level API and retrieve the settings. If mobile application allows their end users to create user or device specific settings, it
uses user storage or device storage APIs. If there are no user level or device level settings, then the GET operation response includes only application level settings. The device storage API merges settings in both Application storage and user storage.

Table 133:

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Storage Service [page 331]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Storage Service [page 334]</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT Storage Service [page 337]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Storage Service [page 343]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Storage Service [page 340]</td>
</tr>
</tbody>
</table>

1.5.1.3.14.1.1 GET Storage Service

Retrieve a mobile data storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>

HTTP Method GET

Request Parameters

Table 134:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application identifier.</td>
<td>String</td>
</tr>
<tr>
<td>OriginName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is “hcpms”.</td>
<td>String</td>
</tr>
<tr>
<td>configurationName</td>
<td>Optional</td>
<td>Configuration name.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>propName</td>
<td>Optional</td>
<td>Property name. The propName, propNameLv2 or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of the properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.</td>
<td>String</td>
</tr>
</tbody>
</table>

#### Note

Any special characters in the configuration name and property name includes must be correctly encoded in URL.

#### Request Example

```plaintext
GET /application/appID1/endpoints/__metadata/readRoles  
   - get the read roles of endpoints configuration.
GET /application/appID1/endpoints/ep1/url  
   - get the url value of ep1 endpoint.
```

#### Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Property value, which may be a JSON object, a primitive value, or any array except for JSON one.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Specified application or property not exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

#### Request (By User Name)

Retrieve user-specific configuration.

**URI**: `https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/< propName>/< propNameLv2>/ < propNameLvN>`
Request Parameters

Table 136:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>user name</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
<tr>
<td>Overwrite</td>
<td>Optional</td>
<td>Whether result needs to merge with application-level data. The default value is true.</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 137:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>User property created.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Specified application or property does not exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Note

Set the overwrite property for a user-level configuration to true, to overwrite an application-level configuration. The current level and its parent level configuration is merged by using the overwrite property.

Request (By Device Name)

Retrieve a device-specific configuration or property.

URI: https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/configurationName/<propName>/<propNameLv2>/<propNameLvN>

Request Parameters

Table 138:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Unique identifier for device type</td>
<td>String</td>
</tr>
<tr>
<td>Overwrite</td>
<td>Optional</td>
<td>Whether result needs to merge with application-level data. The default value is true.</td>
<td>Boolean</td>
</tr>
</tbody>
</table>
### Table 139:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Device property created.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Device property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

### Related Information

Storage Service [page 285]

### 1.5.1.3.14.1.2 POST Storage Service

Create a mobile data storage service based on application configuration, user property, or device property.

#### Request (By Application Configuration)

Create an application configuration, which is a JSON object that includes arbitrary properties. Each configuration can include a "__metadata" property which includes two properties for access control: readRoles and writeRoles.

**URI**

https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>

**HTTP Method** POST

#### Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application identifier</td>
<td>String</td>
</tr>
<tr>
<td>configurationName</td>
<td>Optional</td>
<td>Configuration name</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;hcpms&quot;.</td>
<td>String</td>
</tr>
</tbody>
</table>
**Parameter** | **Type** | **Description** | **Parameter Type**
---|---|---|---
`propName` | Optional | Property name. The `propName`, `propNameLv2`, or `propNameLvN` are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. The property name must exist in the request URL, otherwise, a 404 code for PUT/PATCH/DELETE/GET methods is returned. | String

**Request Example**

This example code creates a configuration for an application named "test1".

```json
POST Storage/v1/runtime/application/test1
{
   "endpoints":{
      "ep1": { "url": "http://endpoint1"},
      "ep2": { "url": "http://endpoint2"}
   },
   "push": {
      "APN": { "host": "www.apple.com",
      "user": "useraaa",
      "pwd": "pwd"},
      "GCM": { "host": "www.google.com",
      "token": "ABABABA"},
      "__metadata": {
         "readRoles": ["Role1", "Role2"],
      "writeRoles": ["Role1"]
   }
}
```

After creating the application configuration, you can use property names to add a new endpoint for the application.

```json
{
   "endpoints":{
      "ep1": { "url": "http://endpoint1"},
      "ep2": { "url": "http://endpoint2"},
      "ep3": { "url": "http://endpoint3"},
   },
   "push": {
      "APN": { "host": "www.apple.com",
      "user": "useraaa",
      "pwd": "pwd"},
      "GCM": { "host": "www.google.com",
      "token": "ABABABA"},
      "__metadata": {
         "readRoles": ["Role1", "Role2"],
      "writeRoles": ["Role1"]
   }
}
```
Response Status and Error Codes

Table 141:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Property created.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By User Name)

Create a user-specific configuration.

**URI:** `https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>`

Request Parameters

Table 142:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>user name</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

Request Example

```
POST /application/appID1/user/ab01/ep1
{ "authenticationType": "NoAuth" }
```

Response Status and Error Codes

Table 143:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Property created.</td>
</tr>
<tr>
<td>403</td>
<td>Cannot access other user configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

**Note**

The user name must be the authenticated user name, otherwise, the service returns error code 403.
Request (By Device Name)

Create a device-specific configuration or property.

Details about the creator of the device configuration is stored internally. The storage service API verifies the creator information before accessing device configuration.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/configuration/<configurationName>/<propName>/propNameLv2/propNameLvN

Request Parameters

Table 144: Parameter Type Description Parameter Type

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Unique identifier for device type</td>
</tr>
</tbody>
</table>

Table 145: Code Description

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Device created.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Note

The authenticated user name implicitly accesses the information; however, the user name and device ID identify the device-level configuration.

1.5.1.3.14.1.3 PUT Storage Service

Update mobile data storage service based on application configuration, user property, or device property.

Request (By application configuration)

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/configurationName/<propName>/propNameLv2/propNameLvN

HTTP Method PUT
Request Parameters

Table 146:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Mandatory</td>
<td>Application identifier.</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Mandatory</td>
<td>Origin from where the application is created. The valid value is “hcpms”.</td>
<td>String</td>
</tr>
<tr>
<td>configurationName</td>
<td>Optional</td>
<td>Configuration name.</td>
<td>String</td>
</tr>
<tr>
<td>propName</td>
<td>Optional</td>
<td>Property name. The propName, propNameLv2 or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.</td>
<td>String</td>
</tr>
</tbody>
</table>

**Note**

It replaces the stored configurations by the one provided in request body.

Response Status and Error Codes

Table 147:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated application configuration successfully.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Specified application or property not exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By User Name)

Update a user-specific configuration or property.

**URI:**

```
https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>
```
Request Parameters

Table 148:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>Mandatory</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 149:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated user property successfully.</td>
</tr>
<tr>
<td>403</td>
<td>Cannot access other user’s configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Specified application or property does not exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By Device Name)

Create a device-specific configuration or property.

Details about the creator of the device configuration is stored internally. The storage service API verifies the creator information before accessing device configuration.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/configurationName/<propName>/

Request Parameters

Table 150:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Unique identifier for device type</td>
<td>String</td>
</tr>
</tbody>
</table>

Table 151:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Device created.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Note

The authenticated user name implicitly accesses the information; however, the user name and device ID identify the device-level configuration.
1.5.1.3.14.1.4 PATCH Storage Service

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

The partial configuration after merged with the existing configuration is stored in the database.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/
<propNameLvN>

HTTP Method PATCH

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application identifier.</td>
<td>String</td>
</tr>
<tr>
<td>prop</td>
<td>Required</td>
<td>Partial property value, which can be part of a JSON object, a primitive value, or any array except a JSON array.</td>
<td>Property</td>
</tr>
<tr>
<td>originName</td>
<td>Mandatory</td>
<td>Origin from where the application is created. The valid value is &quot;hcpms&quot;.</td>
<td>String</td>
</tr>
<tr>
<td>configurationName</td>
<td>Optional</td>
<td>Configuration name.</td>
<td>String</td>
</tr>
</tbody>
</table>
### Request Example

In this example, the following code is in application configuration 'appId1'

```json
{
    "endpoints": {
        "ep2": { "url": "http://newEndpoint2" }
    },
    "push": {
        "BES": { "host": "www.blackberry.com", "user": "useraaa", "pwd": "pwd" }
    }
}
```

After existing and new configuration is merged, the new configuration is stored in the database as:

```json
{
    "endpoints": {
        "ep1": { "url": "http://endpoint1" },
        "ep2": { "url": "http://newEndpoint2" }
    },
    "push": {
        "APN": { "host": "www.apple.com", "user": "useraaa", "pwd": "pwd" },
        "GCM": { "host": "www.google.com", "token": "ABABABA" },
        "BES": { "host": "www.blackberry.com", "user": "useraaa", "pwd": "pwd" },
        "__metadata": {
            "readRoles": ["Role1", "Role2"],
            "writeRoles": ["Role1"]
        }
    }
}
```

### Response Status and Error Codes

Table 153:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated application configuration or property.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>Code</td>
<td>Description</td>
</tr>
<tr>
<td>------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>404</td>
<td>Specified application configuration does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

**Request (By User Name)**

Partially update a user-specific configuration or property.

**URI:**
```
https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/
<propNameLv2>/<propNameLvN>
```

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated user property.</td>
</tr>
<tr>
<td>403</td>
<td>Cannot access other user’s configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error</td>
</tr>
</tbody>
</table>

**Request (By Device Name)**

Partially update a device-specific configuration or property.

**URI:**
```
https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/<configurationName>/<propName>/
<propNameLv2>/<propNameLvN>
```

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Unique identifier for device type</td>
<td>String</td>
</tr>
</tbody>
</table>
Table 157:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated device property.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>409</td>
<td>Property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.1.5 DELETE Storage Service

Delete a mobile data storage service based on application configuration, user property, or device property.

Request (By Application Configuration)

Delete application configuration or property.

**URI**

https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/runtime/application/<applicationId>/global/<configurationName>/<propName>/<propNameLv2>/

**HTTP Method** DELETE

**Request Parameters**

Table 158:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Mandatory</td>
<td>Application identifier</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;hcpms&quot;.</td>
<td>String</td>
</tr>
<tr>
<td>configurationName</td>
<td>Optional</td>
<td>Configuration name</td>
<td>String</td>
</tr>
<tr>
<td>propName</td>
<td>Optional</td>
<td>Property name. The propName, propNameLv2 or propNameLvN are arbitrary in Request URL, as long as the property exists in the configuration or parent property value. If one of properties does not exist, it returns 404 code for PUT/PATCH/DELETE/GET methods.</td>
<td>String</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 159:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Deleted of application configuration or property.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Specified application configuration does not exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By User Name)

Delete a user-specific configuration or property.

URI: https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/currentuser/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>

Request Parameters

Table 160:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>Mandatory</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 161:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Deleted user property successfully.</td>
</tr>
<tr>
<td>403</td>
<td>Cannot access other user’s configuration.</td>
</tr>
<tr>
<td>404</td>
<td>User property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error</td>
</tr>
</tbody>
</table>

Request (By Device Name)

Delete a device-specific configuration or property.

URI: https://<HMC base URL>/mobileservices/origin/<originname>/Storage/v1/runtime/application/<applicationId>/device/<deviceId>/<configurationName>/<propName>/<propNameLv2>/<propNameLvN>

344 PUBLIC
Request Parameters

Table 162:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Mandatory</td>
<td>Unique identifier for device type</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 163:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Updated device property.</td>
</tr>
<tr>
<td>403</td>
<td>No permission to access a configuration.</td>
</tr>
<tr>
<td>404</td>
<td>Specified application or device property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.2 Admin APIs

Admin APIs are used to configure settings at application storage level, and manages all settings in both user storage and device storage.

The Admin API can only be accessed by origin authentication URL pattern. It supports two authentication types: Basic, App2App.

**Note**

Currently, only SAP Cloud Platform mobile service for development and operations originated applications are supported.

Table 164:

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Storage Services [page 347]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Storage Services [page 346]</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT Storage Services [page 349]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Storage Services [page 353]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Storage Services [page 351]</td>
</tr>
</tbody>
</table>
1.5.1.3.14.2.1 POST Storage Services

Add new entity to application, user, and device storage sets.

Request (By Application Storage Set)

Create a new entity to ApplicationStorageSet

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet

HTTP Method POST

Response Status and Error Codes

Table 165:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Entity created.</td>
</tr>
<tr>
<td>409</td>
<td>Entity already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By User Storage Set)

Create a new entity in user storage set.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet

Response Status and Error Codes

Table 166:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Entity created.</td>
</tr>
<tr>
<td>409</td>
<td>Entity already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Request (By Device Storage Set)

Create a new entity in device storage set.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet
Response Status and Error Codes

Table 167:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Device entity created.</td>
</tr>
<tr>
<td>409</td>
<td>Entity already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.2.2 GET Storage Services

Retrieve new entities from application, user, and device storage sets.

Request (By Application Storage Set)

Retrieve an entity from application storage set by key.

**URI**

```
https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet/(ApplicationId='<ApplicationId>')
```

Retrieve all the entities from application storage.

**URI**

```
https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet?$<system_query_Option>
```

**Note**

Supported OData system query options are $count, $skip, $top.

**HTTP Method** **GET**

Request Parameters

Table 168:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Key: application ID</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;hcpms&quot;.</td>
<td>String</td>
</tr>
<tr>
<td>$top</td>
<td>Optional</td>
<td>Show only the first n elements.</td>
<td>Integer</td>
</tr>
<tr>
<td>$skip</td>
<td>Optional</td>
<td>Skip the first n elements.</td>
<td>Integer</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>-----------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>$count</td>
<td>Optional</td>
<td>Allows users to request a count of the matching resources. The number is included with the resources in the response.</td>
<td>Boolean</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 169:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set found in application storage set.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
</tbody>
</table>

Request (By User Storage Set)

Retrieve an entity from user storage set by key.

**URI:** https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId='ApplicationId', Username='username')

Retrieve all entities from user storage set.

**URI:** https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet?$<system_query_Option>

Request Parameters

Table 170:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 171:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set created.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Request (By Device Storage Set)

Retrieve a entity from device storage set by key.
URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/
DeviceStorageSet(ApplicationId='<ApplicationId>', Username='<username>',
DeviceId='<deviceId>')

Retrieve all entities from device storage set.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/
DeviceStorageSet?$<system_query_Option>

Request Parameters

Table 172:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Unique identifier for device type</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 173:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set created.</td>
</tr>
<tr>
<td>404</td>
<td>Device property already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.2.3 PUT Storage Services

Update entities from application, user, and device storage sets.

Request (By Application Storage Set)

Update entity in application storage set.

URI https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/
ApplicationStorageSet(''<ApplicationId>'')

HTTP Method PUT

Request Parameters

Table 174:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>application ID</td>
<td>String</td>
</tr>
</tbody>
</table>
### Parameter Type

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;hcpms&quot;.</td>
<td>String</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

**Table 175:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity updated in application storage set.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
</tbody>
</table>

### Request (By User Storage Set)

Update entities in user storage set.

**URI:**

`https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId='<ApplicationId>', Username='<username>')`

### Request Parameters

**Table 176:**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

**Table 177:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set updated in user storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

### Request (By Device Storage Set)

Update entity in device storage set.

**URI:**

`https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId='<ApplicationId>', Username='<username>', DeviceId='<deviceId>')`
Request Parameters

Table 178:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Device ID</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 179:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity updated in device storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.2.4 PATCH Storage Services

Merge a mobile data in the existing storage service based on application configuration, user property, or device property.

Request (By Application Storage Set)

Update entity in application storage set.

**URI**

https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet(<ApplicationId>)

**HTTP Method** PATCH

Request Parameters

Table 180:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>application ID</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is “hcpms”.</td>
<td>String</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 181:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity updated in application storage set.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist</td>
</tr>
</tbody>
</table>

Request (By User Storage Set)

Update entities in user storage set.

**URI:**

https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/
UserStorageSet(ApplicationId='<ApplicationId>', Username='<username>')</n>

Request Parameters

Table 182:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 183:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set is updated in the user storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Request (By Device Storage Set)

Update entity in device storage set.

**URI:**

https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/
DeviceStorageSet(ApplicationId='<ApplicationId>', Username='<username>',
DeviceId='<deviceId>' )

Request Parameters

Table 184:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Device ID</td>
<td>String</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 185:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity set is updated in the device storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.14.2.5 DELETE Storage Services

Delete entities from application, user, and device storage sets.

Request (By Application Storage Set)

Delete entity from application storage set.

URI: https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/ApplicationStorageSet('<ApplicationId>')

HTTP Method DELETE

Request Parameters

Table 186:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>application ID</td>
<td>String</td>
</tr>
<tr>
<td>originName</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is “hcpms”.</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 187:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Entity deleted from application storage set.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
</tbody>
</table>
Request (By User Storage Set)

Delete entities from user storage set.

**URI:** https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/UserStorageSet(ApplicationId=''<ApplicationId>'', Username=''<username>'')

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>username</td>
<td>Required</td>
<td>User name</td>
<td>String</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Entity set is deleted from the user storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Request (By Device Storage Set)

Delete entities from device storage set.

**URI:** https://<HMC base URL>/mobileservices/origin/<originName>/Storage/v1/admin/DeviceStorageSet(ApplicationId=''<ApplicationId>'', Username=''<username>'', DeviceId=''<deviceId>'')

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>Device ID</td>
<td>String</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Entity set is deleted in the device storage set.</td>
</tr>
<tr>
<td>404</td>
<td>Entity does not exist.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
1.5.1.3.15 Client Resources Service

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application. The uploaded logs can be viewed from SAP Cloud Platform mobile service for development and operations Admin Cockpit.

Client Resources service supports two types of APIs:

- Runtime APIs
- Admin APIs

Related Information

Runtime APIs [page 355]
Admin APIs [page 358]

1.5.1.3.15.1 Runtime APIs

The Client Resource Bundle API allows the application to download resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

Table 192: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Client Resources Service [page 355]</td>
</tr>
</tbody>
</table>

1.5.1.3.15.1.1 GET Client Resources Service

HTTP Method: GET

Default Resource Bundle

This method allows you to download the default resource bundle an application.

URL: https://<HMC base URL>/mobileservices/application/{applicationId}/bundles/v1/runtime/bundle/application/{applicationId}

Request Parameters
Table 193:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application ID</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 194:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Resource bundle downloaded successfully.</td>
</tr>
<tr>
<td></td>
<td>X-BUNDLE-EXTENSION - The file extension as a response header.</td>
</tr>
<tr>
<td>404</td>
<td>Application or resource bundle not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

Specified Resource Bundle

This method allows you to download the specified resource bundle of an application.

**URL:**
https://<HMC base URL>/mobileservices/application/{applicationId}/bundles/v1/runtime/bundle/application/{applicationId}/bundle/{bundlename}/version/{version}

Request Parameters

Table 195:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application ID</td>
<td>string</td>
</tr>
<tr>
<td>bundlename</td>
<td>Required</td>
<td>Bundle name in the format bundlename:version.</td>
<td>string</td>
</tr>
<tr>
<td>version</td>
<td>Required</td>
<td>Bundle version in the format bundlename:version.</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 196:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Resource bundle downloaded successfully.</td>
</tr>
<tr>
<td></td>
<td>X-BUNDLE-EXTENSION - The file extension as a response header.</td>
</tr>
<tr>
<td>404</td>
<td>Application or resource bundle not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

List All Resource Bundles

This method allows you to list all the resource bundles of an application.
**Request Parameters**

Table 197:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application ID.</td>
<td>string</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

Table 198:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Resource bundles listed successfully.</td>
</tr>
<tr>
<td>404</td>
<td>Application not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unknown error.</td>
</tr>
</tbody>
</table>

**Downloaded the Latest Resource Bundle**

This method allows you to download the latest resource bundle of an application by ordering bundle versions lexicographically.

**Example**

- 1.1.2 is a later version than 1.1.1
- 1.1.2 is a later version than 1.1.1.a
- 1.2.a.1 is a later version than 1.1.1.a
- 1.a is a later version than 1.2.a.1

**Request Parameters**

Table 199:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>If-None-Match</td>
<td>Optional</td>
<td>The etag value returned during the last GET latest resource bundle request</td>
<td>string</td>
</tr>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application ID</td>
<td>string</td>
</tr>
<tr>
<td>bundlename</td>
<td>Required</td>
<td>Bundle name</td>
<td>string</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 200:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
</table>
| 200  | Resource bundle downloaded successfully:  
  - etag - etag value of downloaded resource bundle.  
  - X-BUNDLE-EXTENSION - the file extension as a response header. |
| 304  | Latest resource bundle of an application. |
| 404  | Application or resource bundle not found. |
| 500  | Unknown error. |

1.5.1.3.15.2 Admin APIs

The Client Resource Bundle API allows the application to download the resources from the server. The Resource file is uploaded by the Administrator, and then downloaded by using the application.

Table 201: Admin API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Client Resources Service [page 358]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Client Resources Service [page 361]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Client Resources Service [page 362]</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT Client Resources Service [page 363]</td>
</tr>
</tbody>
</table>

1.5.1.3.15.2.1 GET Client Resources Service

HTTP Method: GET

Service Document

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin

Response Status and Error Codes

Table 202:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Service document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
Metadata Document

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata

Response Status and Error Codes

Table 203:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Metadata document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Application Resource Bundles

This method allows you to get the entities from the Entity Set Application Resource Bundles.

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles

Response Status and Error Codes

Table 204:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet ApplicationResourceBundles.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Application Resource Bundles by Key

This method allows you to get the entity from the Application Resource Bundles by key, and it returns the entity with the key from the Application Resource Bundles.

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')

Request Parameters

Table 205:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationId</td>
<td>Required</td>
<td>key: ApplicationId</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
</tr>
<tr>
<td>BundleName</td>
<td>Required</td>
<td>key: BundleName</td>
</tr>
</tbody>
</table>
### Binary Content of the Application Resource

This method allows you to get the binary content of the application resource.

**URL:** `https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/$value`

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationId</td>
<td>Required</td>
<td>key: ApplicationId</td>
<td>string</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
<tr>
<td>BundleName</td>
<td>Required</td>
<td>key: BundleName</td>
<td>string</td>
</tr>
<tr>
<td>BundleVersion</td>
<td>Required</td>
<td>key: BundleVersion</td>
<td>string</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

**Table 208:**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The binary content of the application resource.</td>
</tr>
<tr>
<td>404</td>
<td>Entity Not Found.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
1.5.1.3.15.2.2 POST Client Resources Service

HTTP Method: POST

Application Resource Bundles

This method allows you to add a new entity to the Entity Set Application Resource Bundles. The uploaded logs can be viewed from SAP Cloud Platform mobile service for development and operations Admin Cockpit.

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles

Request Parameters

Table 209:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationResourceBundle</td>
<td>Optional</td>
<td>The new entity.</td>
<td>• ApplicationId: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• BundleName: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• BundleVersion: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• BundleExtension: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• IsDefault: boolean</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is “SAP Cloud Platform mobile service for development and operations”</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 210:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created entity.</td>
</tr>
<tr>
<td>409</td>
<td>Entity already exists.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Set Default

This method allows you to invoke the action Set Default.

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/com.sap.mobile.server.bundle.admin.v1.SetDefault

Request Parameters
Table 211:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationId</td>
<td>Required</td>
<td>key: ApplicationId</td>
<td>string</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
<tr>
<td>BundleName</td>
<td>Required</td>
<td>key: BundleName</td>
<td>string</td>
</tr>
<tr>
<td>BundleVersion</td>
<td>Required</td>
<td>key: BundleVersion</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 212:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Entity Not Found.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.15.2.3 DELETE Client Resources Service

This method allows you to delete an entity from the Application Resource Bundles by using the key.

HTTP Method: DELETE

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')

Request Parameters

Table 213:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationId</td>
<td>Required</td>
<td>key: ApplicationId</td>
<td>string</td>
</tr>
<tr>
<td>Parameter</td>
<td>Type</td>
<td>Description</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>------------</td>
<td>-----------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------------</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
<tr>
<td>BundleName</td>
<td>Required</td>
<td>key: BundleName</td>
<td>string</td>
</tr>
<tr>
<td>BundleVersion</td>
<td>Required</td>
<td>key: BundleVersion</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 214:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet ApplicationResourceBundles.</td>
</tr>
<tr>
<td>404</td>
<td>Entity Not Found.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.15.2.4 PUT Client Resources Service

This method allows you to set the binary content of the application resource.

**HTTP Method:** PUT

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')/$value

**Request Parameters**

Table 215:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationId</td>
<td>Required</td>
<td>key: ApplicationId</td>
<td>string</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
<tr>
<td>BundleName</td>
<td>Required</td>
<td>key: BundleName</td>
<td>string</td>
</tr>
</tbody>
</table>
### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>BundleVersion</td>
<td>Required</td>
<td>key: BundleVersion</td>
<td>string</td>
</tr>
<tr>
<td>resourceBinary</td>
<td>Optional</td>
<td>The binary content of the application resource.</td>
<td>binary</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

Table 216:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Set the binary content of the application resource successfully.</td>
</tr>
<tr>
<td>404</td>
<td>Entity Not Found.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

### 1.5.1.3.16 Client Log Upload Service

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

Client Log Upload service supports two types of APIs:

- Runtime APIs
- Admin APIs

### Related Information

- Runtime APIs [page 364]
- Admin APIs [page 366]

### 1.5.1.3.16.1 Runtime APIs

The Client Log Upload API allows the application to upload the client logs to the server for further analysis.

Table 217: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>POST Client Log Upload Service [page 365]</td>
</tr>
</tbody>
</table>
1.5.1.3.16.1.1 POST Client Log Upload Service

The Client Log Upload API allows the application to upload the client logs to the server for further analysis. This method allows you to upload client logs for an application. If the client log upload option is enabled for the specified application, then the client logs get uploaded.

HTTP Method: POST.

URL: https://<HMC base URL>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}

Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>Application identifier</td>
<td>String</td>
</tr>
<tr>
<td>deviceId</td>
<td>Optional</td>
<td>Device identifier. Its value will be saved with uploaded log if specified.</td>
<td>String</td>
</tr>
<tr>
<td>clientlogs</td>
<td>Required</td>
<td>Client logs. It can be zipped data, multipart data or simple binary data. Log format: ~~~#Date time#Severity#CorrelationId#Source#Location#Message# ~~~ The format of Date time is &quot;YYYY-MM-DDThh:mm:ss.sTZD&quot;, such as &quot;2014-07-01T17:16:08.637+02:00&quot;. The valid values of Severity are DEBUG, INFO, WARN, ERROR and FATAL. The CorrelationId value is optional, and can be left as empty if client cannot provide it for each log message. The Message value is log message generated by client, and # character in it must be escaped to avoid corrupt uploading.</td>
<td></td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 219:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Client logs uploaded.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid log format.</td>
</tr>
<tr>
<td>403</td>
<td>Client log upload is not enabled for the application.</td>
</tr>
<tr>
<td>404</td>
<td>No application found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.16.2 Admin APIs

The Admin API is used for managing the Log Upload policy. SAP Cloud Platform mobile service for development and operations logging Admin API provides a unique interface for all the logs, and it allows you to get the uploaded client logs.

Table 220: Admin API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Client Log Upload Service [page 366]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Client Log Upload Service [page 368]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Client Log Upload Service [page 368]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Client Log Upload Service [page 369]</td>
</tr>
</tbody>
</table>

1.5.1.3.16.2.1 GET Client Log Upload Service

HTTP Method: GET

Service Document

URL: https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin

Response Status and Error Codes

Table 221:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Service document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
Metadata Document

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/$metadata

**Response Status and Error Codes**

Table 222:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Metadata document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

**Entity Set Application Policy**

This method allows you to get the entities from the Entity Set Application Policy.

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet

**Response Status and Error Codes**

Table 223:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet ApplicationPolicy</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

**Application Policy by Key**

This method allows you to get the entity from the Application Policy by key, and it returns the entity with the key from the Application Policy.

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')

**Request Parameters**

Table 224:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 225:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet ApplicationPolicy.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.16.2.2 POST Client Log Upload Service

This method allows you to post the new entity to the Entity Set Application Policy.

HTTP Method: **POST**

URL: `https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet`

Table 226:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationPolicy</td>
<td>Required</td>
<td>The new entity</td>
<td>• ApplicationId: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• LogUploadEnabled: boolean</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created. The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 227:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created entity.</td>
</tr>
<tr>
<td>409</td>
<td>Client log upload policy already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.16.2.3 PATCH Client Log Upload Service

This method allows you to update the entity in the Entity Set Application Policy.

HTTP Method: **PATCH**

URL: `https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')`
Request Parameters

Table 228:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
<tr>
<td>ApplicationPolicy</td>
<td>Required</td>
<td>The entity to patch.</td>
<td>● ApplicationId: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● LogUploadEnabled: boolean</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created.</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 229:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.16.2.4 DELETE Client Log Upload Service

This method allows you to delete an entity in the Entity Set Application Policy.

HTTP Method: **DELETE**

**URL:** `https://<HMC base URL>/mobileservices/origin/{origin}/clientlogs/v1/admin/ApplicationPolicySet('{applicationId}')`

Request Parameters

Table 230:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
<tr>
<td>Origin</td>
<td>Required</td>
<td>Origin from where the application is created.</td>
<td>string</td>
</tr>
<tr>
<td></td>
<td></td>
<td>The valid value is &quot;SAP Cloud Platform mobile service for development and operations&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

SAP Cloud Platform Mobile Service for Development and Operations
SAP Cloud Platform Mobile Service for Development and Operations

PUBLIC 369
Table 231: Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.17 Push as API Service

The Push as API service allows application developers to push updates from the back-end data source to applications that are running on mobile devices.

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

The Push as API Service also allows you to send SMS notifications. You can send an SMS notification to the application instead of a native push notification. The Push as API service allows you to use both SMS notifications as well as native push notifications.

The Push as API service supports the following types of APIs:

- Runtime APIs
- Admin APIs
- Back-end APIs

Related Information

Runtime APIs [page 370]
Admin APIs [page 378]
Back-end APIs [page 383]

1.5.1.3.17.1 Runtime APIs

The mobile services API can be consumed by mobile applications running on a device, server applications hosted on HCP, and any other back-end applications that can access the API over the internet.

The Push as API service provides services for mobile devices to register for the push service of a specific application in order to receive notifications, as well as the services to manage the lifecycle of their registration.
When registering a device it is now possible to specify a phone number (MSISDN) in the register device request payload:

```json
{
    "deviceModel": "Android",
    "msisdn": "+491234567890"
}
```

- Make sure that the phone number is in an international format (+, country code, network code, number), and is stored along the device registration.
- You can register a device for both SMS notification and native push.
- Example of registering a device for SMS notification and GCM:

```json
Example
{
    "deviceModel": "Android",
    "pushToken": "abcdefghijklmnopqrstuvxyz1234567890",
    "msisdn": "+491234567890"
}
```

When you use JSON Web Token (JWT) authentication, the sap.mob.roles element containing the role should be a part of the JWT. For example, to access a runtime service, the JWT payload is {"sub": "P1940703319", "sap.mob.roles": ["Notification User"], "iss": "hcpms"}.

Table 232: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Push as API Service [page 371]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Push as API Service [page 372]</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT Push as API Service [page 374]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Push as API Service [page 377]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Push as API Service [page 378]</td>
</tr>
</tbody>
</table>

### 1.5.1.3.17.1.1 GET Push as API Service

Enables a device to retrieve an application ID or a sender ID. For example, you could use the GET method to query for the Google Cloud Messaging (GCM) sender ID before registering a device.

**URI:** https://<HMC_base_URL>/mobileservices/push/v1/runtime/applications/{applicationId}/pushconfigurations/os/{operatingSystem}/pushid

**HTTP Method** GET
### Request Parameters

Table 233:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>operatingSystem</td>
<td>Device Type</td>
<td>Yes</td>
<td>String; (windows</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

Table 234:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Push ID returned.</td>
</tr>
<tr>
<td>400</td>
<td>Operating system not supported.</td>
</tr>
<tr>
<td>404</td>
<td>Application not found.</td>
</tr>
<tr>
<td>500</td>
<td>Cannot load push sender ID.</td>
</tr>
</tbody>
</table>

### 1.5.1.3.17.1.2 POST Push as API Service

Uploads push tokens to HCPmsdo.

The Push endpoint lets you register push tokens, which deliver native push notifications to devices. Devices must indicate the platform they run on, to enable connection to the correct notification delivery service.

**Note**

This service overrides any existing device registration push data information. The existing values, which are not set in the updateDeviceRegistrationPushData element, are reset.

### Request (by Device)

**URI:** https://<HMC base URL>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices

**HTTP Method** POST
Request Parameters

Table 235:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>registerDevicePushData</td>
<td>Enables native push delivery.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>operatingSystem</td>
<td>Device Type</td>
<td>Yes</td>
<td>String; (ios</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 236:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Registration successful.</td>
</tr>
<tr>
<td>400</td>
<td>Incomplete request.</td>
</tr>
<tr>
<td>409</td>
<td>Device ID already exists for the given application.</td>
</tr>
<tr>
<td>500</td>
<td>Registration failed.</td>
</tr>
</tbody>
</table>

Request (by Device ID)

URI: https://<HMC base URL>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}

HTTP Method POST

Request Parameters

Table 237:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>registerDeviceByIdPushData</td>
<td>Enables native push delivery.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Required</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>operatingSystem</td>
<td>Device Type</td>
<td>Yes</td>
<td>String; (ios</td>
</tr>
<tr>
<td>deviceId</td>
<td>Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 238:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Registration successful.</td>
</tr>
<tr>
<td>400</td>
<td>Incomplete request.</td>
</tr>
<tr>
<td>409</td>
<td>Device ID already exists for the given application.</td>
</tr>
<tr>
<td>500</td>
<td>Registration failed.</td>
</tr>
</tbody>
</table>

1.5.1.3.17.1.3 PUT Push as API Service

Updates an existing push token.

Request (by Device ID)

URI: https://<HMC base URL>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}

HTTP Method **PUT**

Request Parameters

Table 239:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>updateDeviceRegistrationPushData</td>
<td>Enables native push delivery.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Required</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>operatingSystem</td>
<td>Device Type</td>
<td>Yes</td>
<td>String; (ios</td>
</tr>
<tr>
<td>deviceId</td>
<td>Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 240:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Registration update successful.</td>
</tr>
<tr>
<td>400</td>
<td>The request does not contain all required information.</td>
</tr>
<tr>
<td>404</td>
<td>Registration not found for the given Device ID.</td>
</tr>
<tr>
<td>500</td>
<td>Registration update failed.</td>
</tr>
</tbody>
</table>

Updates the status of an existing notification. This service can be called by connected client devices and two status’ - received and consumed are allowed.

Request (by Application ID)

URI: https://<HMC base URL>/mobileservices/push/v1/runtime/applications/{applicationId}/notifications/{notificationId}/status/{status}

Updates the status of an existing notification. Connected client devices are allowed to call only the received or consumed status respectively.

HTTP Method PUT

Request Parameters

Table 241:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>An application ID is required for authentication.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
<td>Required</td>
<td>Parameter Type</td>
</tr>
<tr>
<td>-------------</td>
<td>------------------------------------------------------------------</td>
<td>----------</td>
<td>----------------</td>
</tr>
<tr>
<td>notificationId</td>
<td>The ID of the notification for which the status has to be updated.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>status</td>
<td>The new status which needs to be set, only received and consumed are allowed.</td>
<td>Yes</td>
<td>String; (received</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 242:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Registration update successful.</td>
</tr>
<tr>
<td>400</td>
<td>The request does not contain all required information.</td>
</tr>
<tr>
<td>404</td>
<td>Notification not found for the given ID.</td>
</tr>
</tbody>
</table>

Request (by Notification ID)

URI: `https://<HMC base URL>/mobileservices/push/v1/runtime/notifications/{notificationId}/status/{status}`

HTTP Method **PUT**

Request Parameters

Table 243:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>notificationId</td>
<td>The ID of the notification for which the status has to be updated.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>status</td>
<td>The new status which needs to be set, only received and consumed are allowed.</td>
<td>Yes</td>
<td>String; (received</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 244:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Registration update successful.</td>
</tr>
<tr>
<td>400</td>
<td>The request does not contain all required information.</td>
</tr>
</tbody>
</table>
### 1.5.1.3.17.1.4 PATCH Push as API Service

This method allows you to merge push tokens. If no device registration exists for the given device ID, the registration is created on the fly.

**URI:** `https://<HMC base URL>/mobileservices/push/v1/runtime/applications/{applicationId}/os/{operatingSystem}/devices/{deviceId}`

**HTTP Method** `PATCH`

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>mergeDeviceRegistrationPushData</td>
<td>Enables native push delivery.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>operatingSystem</td>
<td>Device Type</td>
<td>Yes</td>
<td>String; (ios</td>
</tr>
<tr>
<td>deviceId</td>
<td>Device identifier such as the IMSI, MSISDN, SIM, or a similar identifier that is unique to the device. This is optional and can be created if it is not provided.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

#### Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Registration merged.</td>
</tr>
<tr>
<td>400</td>
<td>The request does not contain all required information.</td>
</tr>
<tr>
<td>500</td>
<td>Registration update failed.</td>
</tr>
</tbody>
</table>
1.5.1.3.17.1.5 DELETE Push as API Service

Delete a native push configuration.

URI: https://<HMC base URL>mobileservices/origin/hcpms/push/v1/admin/applications/{applicationId}/pushconfigurations

HTTP Method DELETE

Request Parameters

Table 247:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>The ID of the current application. This should match the application ID used while configuring push on HCPms for your app.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 248:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Push configuration successfully deleted.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid request.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for the application.</td>
</tr>
<tr>
<td>404</td>
<td>Application not found.</td>
</tr>
<tr>
<td>500</td>
<td>Push configuration deletion failed.</td>
</tr>
</tbody>
</table>

1.5.1.3.17.2 Admin APIs

The Administrator role is required to access Admin APIs.

Push as API service provides services to manage the push configurations on at the application level. It also supports all the major push providers, such as APNS, GCM, WNS, and it can be configured individually. The existing REST and OData Admin services supports SMS, which integrates all the relevant configuration options. You can configure the SMS notification similar to the native push configuration. An example request payload to configure an application for SMS looks like the following

```json
{
  "gcm": {
    "apiKey": "12341234",
    "senderId": "abc"
  },
  "sms": {
```

The SMS configuration is just an additional configuration property that can be used along native push configurations. You can also update the configuration by using the corresponding PUT and PATCH requests.

Table 249: Admin API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Push as API Service [page 379]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Push as API Service [page 380]</td>
</tr>
<tr>
<td>PUT</td>
<td>PUT Push as API Service [page 380]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Push as API Service [page 381]</td>
</tr>
</tbody>
</table>

1.5.1.3.17.2.1 GET Push as API Service

Retrieves native push configuration based on application ID.

URI: https://<HMC base URL>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations

HTTP Method GET

Request Parameters

Table 250:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 251:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Push configuration successfully retrieved</td>
</tr>
<tr>
<td>400</td>
<td>Invalid request.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for application.</td>
</tr>
</tbody>
</table>
1.5.1.3.17.2.2 POST Push as API Service

With a valid push configuration application registrations can be created as and when required using the POST method.

**URI:** https://<HMC base URL>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations

**HTTP Method** POST

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>The push configuration.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>The application has been successfully configured for push.</td>
</tr>
<tr>
<td>304</td>
<td>A push configuration for this application already exists</td>
</tr>
<tr>
<td>400</td>
<td>Invalid push configuration.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for application.</td>
</tr>
<tr>
<td>500</td>
<td>Configuration not created.</td>
</tr>
</tbody>
</table>

1.5.1.3.17.2.3 PUT Push as API Service

Updates the native push configuration.

**URI:** https://<HMC base URL>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations

**HTTP Method** PUT
### Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>The push configuration.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Push configuration successfully updated.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid push configuration.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for application.</td>
</tr>
<tr>
<td>404</td>
<td>Application or push configuration not found.</td>
</tr>
<tr>
<td>500</td>
<td>Application configuration failed.</td>
</tr>
</tbody>
</table>

### 1.5.1.3.17.2.4 PATCH Push as API Service

Updates an application’s native push configuration. If the provided application does not exist, an application can be created and registered on the fly. This method merges a provided push configuration with any that already exist for an application.

**URI:** `https://<HMC base URL>/mobileservices/push/v1/admin/applications/{applicationId}/pushconfigurations`

**HTTP Method:** `PATCH`

### Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>config</td>
<td>The push configuration.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>The ID of the current application which should match the ID used for configuring your app for push notifications.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 257:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Push configuration successfully updated.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid push configuration.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for application.</td>
</tr>
<tr>
<td>404</td>
<td>Application or push configuration not found.</td>
</tr>
<tr>
<td>500</td>
<td>Configuration not created.</td>
</tr>
</tbody>
</table>

1.5.1.3.17.2.5 DELETE Push as API Service

Delete a native push configuration.

**URI:**
https://<HMC base URL>mobileservices/origin/hcpms/push/v1/admin/applications/{applicationId}/pushconfigurations

**HTTP Method** *DELETE*

Request Parameters

Table 258:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>The ID of the current application. This should match the application ID used while configuring push on HCPms for your app.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 259:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Push configuration successfully deleted.</td>
</tr>
<tr>
<td>400</td>
<td>Invalid request.</td>
</tr>
<tr>
<td>403</td>
<td>Access denied for the application.</td>
</tr>
<tr>
<td>404</td>
<td>Application not found.</td>
</tr>
<tr>
<td>500</td>
<td>Push configuration deletion failed.</td>
</tr>
</tbody>
</table>
1.5.1.3.17.3 Back-end APIs

The notification user role is required to access back-end APIs.

The Push as API service provides services for the backend applications to trigger push notifications for either specific devices, or all devices of an application, or all devices of a certain user.

The Push as API service also allows you to send SMS notifications. You can send an SMS notification to the application instead of a native push notification. The Push as an API service allows you to use both SMS notifications as well as native push notifications.

Table 260: Back-end API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>POST Push as API Service [page 383]</td>
</tr>
<tr>
<td>GET</td>
<td>GET Push as API Service [page 386]</td>
</tr>
</tbody>
</table>

1.5.1.3.17.3.1 POST Push as API Service

Triggers a native push to users, an application, or a capability based on the request.

Request (by User)

**URI:** https://<HMC base URL>/mobileservices/push/v1/backend/applications/{applicationId}/notifications/users

**HTTP Method:** POST

Request Parameters

Table 261:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pushToUsersPayload</td>
<td>Target users</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>Target application ID</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 262:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The push notification sent successfully</td>
</tr>
<tr>
<td>202</td>
<td>The push request has been successfully accepted.</td>
</tr>
</tbody>
</table>
### Code and Description

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>304</td>
<td>The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.</td>
</tr>
<tr>
<td>400</td>
<td>The push request is syntactically incorrect.</td>
</tr>
<tr>
<td>500</td>
<td>Push failed.</td>
</tr>
</tbody>
</table>

### Request (by Notifications)

You can trigger a native push to a specified device used by the given user. This back-end API can provide a device ID in addition to the username in case the push message should be received only on a certain device and not to all the devices linked to the given user.

**URL:** https://<HMC base URL>/mobileservices/push/v1/backend/applications/{applicationId}/users/{username}/devices/{deviceId}/notifications

**HTTP Method** POST

### Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pushPayload</td>
<td>The push notification payload.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>Target application ID.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>username</td>
<td>The user to whom the device is assigned.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>deviceId</td>
<td>The device ID of the device receiving the push notification.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The push notification sent successfully</td>
</tr>
<tr>
<td>202</td>
<td>The push request has been successfully accepted.</td>
</tr>
<tr>
<td>304</td>
<td>The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.</td>
</tr>
<tr>
<td>400</td>
<td>The push request is syntactically incorrect.</td>
</tr>
<tr>
<td>500</td>
<td>Push failed.</td>
</tr>
</tbody>
</table>
Request (by Application)

**URI:** https://<HMC base URL>/mobileservices/push/v1/backend/applications/{applicationId}/notifications

**HTTP Method** POST

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pushToApplicationPayload</td>
<td>Notification payload.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>applicationId</td>
<td>Target application ID.</td>
<td>Yes</td>
<td>String; Integer</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The push notification sent successfully</td>
</tr>
<tr>
<td>202</td>
<td>The push request has been successfully accepted.</td>
</tr>
<tr>
<td>304</td>
<td>The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.</td>
</tr>
<tr>
<td>400</td>
<td>The push request is syntactically incorrect.</td>
</tr>
<tr>
<td>500</td>
<td>Push failed.</td>
</tr>
</tbody>
</table>

Request (by Capability)

**URI:** https://<HMC base URL>/mobileservices/push/v1/backend/capabilities/{capabilityName}/notifications

**HTTP Method** POST

**Request Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>pushToCapabilitiesPayload</td>
<td>A list of capability users and a notification object.</td>
<td>Yes</td>
<td>String</td>
</tr>
<tr>
<td>capabilityName</td>
<td>Target capability.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 268:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The push notification sent successfully</td>
</tr>
<tr>
<td>202</td>
<td>The push request has been successfully accepted.</td>
</tr>
<tr>
<td>304</td>
<td>The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.</td>
</tr>
<tr>
<td>400</td>
<td>The push request is syntactically incorrect.</td>
</tr>
<tr>
<td>500</td>
<td>Push failed.</td>
</tr>
</tbody>
</table>

1.5.1.3.17.3.2 GET Push as API Service

Fetches the status of a push notification.

**URI:** `https://<HMC base URL>/mobileservices/push/v1/backend/notifications/{notificationId}/status`

**HTTP Method** `GET`

**Request Parameters**

Table 269:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Required</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>notificationId</td>
<td>Push notification ID.</td>
<td>Yes</td>
<td>String</td>
</tr>
</tbody>
</table>

**Response Status and Error Codes**

Table 270:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Push notification status.</td>
</tr>
<tr>
<td>304</td>
<td>The push request could not be sent due to an error in the push payload parameters, which is preventing the server from finding the device registration.</td>
</tr>
<tr>
<td>400</td>
<td>The request is missing its notification ID.</td>
</tr>
<tr>
<td>404</td>
<td>No push notification for the given ID found.</td>
</tr>
<tr>
<td>500</td>
<td>Push notification lookup failed.</td>
</tr>
</tbody>
</table>
1.5.1.3.18 Document Service

The Document Service API allows you to access Document Service repository through SAP Cloud Platform mobile service for development and operations. By using this API, you can consume the Document Service without performing the registration.

Document Service supports two types of APIs:

- Runtime APIs
- Admin APIs

Related Information

Runtime APIs [page 387]
Admin APIs [page 388]

1.5.1.3.18.1 Runtime APIs

Document Service API allows you to access Document Service repository through SAP Cloud Platform mobile service for development and operations. This page describes the Document Service Runtime API of SAP Cloud Platform mobile service for development and operations.

Table 271: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Document Service API [page 387]</td>
</tr>
</tbody>
</table>

1.5.1.3.18.1.1 GET Document Service API

This method allows you to get the metadata of the document repository of {applicationId}. The corresponding security configuration of the application {application} is found and the authentication is performed.

HTTP Method: GET

Metadata Document

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata

Response Status and Error Codes

Request Parameters
### Admin APIs

Document Service API allows you to access Document Service repository through SAP Cloud Platform mobile service for development and operations. This page describes the Document Service Admin API of SAP Cloud Platform mobile service for development and operations.

#### GET Document Service API

**HTTP Method:** GET

**Service Document**

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin

**Response Status and Error Codes**

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Service document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

**Metadata Document**

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/$metadata
Response Status and Error Codes

Table 276:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Metadata document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.18.2.2 POST Document Service API

This method allows you to create a Document Service repository.

HTTP Method: POST.

URL: https://<HMC base URL>/mobileservices/application/{application}/clientlogs/v1/runtime/log/application/{applicationId}?deviceId={deviceId}

Request Parameters

Table 277:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>CmisReposito ry</td>
<td>Optional</td>
<td>A new entity</td>
<td>ApplicationId: string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 278:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created entity successfully</td>
</tr>
<tr>
<td>409</td>
<td>Document repository already exists</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.18.2.3 DELETE Document Service API

This method allows you to delete an entity in EntitySet CmisRepository.

HTTP Method: DELETE

URL: https://<HMC base URL>/mobileservices/origin/{origin}/bundles/v1/admin/ApplicationResourceBundles(ApplicationId='{ApplicationId}',BundleName='{BundleName}',BundleVersion='{BundleVersion}')
### Table 279:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

#### Table 280:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>202</td>
<td>Empty Response</td>
</tr>
<tr>
<td>404</td>
<td>Not Found</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error</td>
</tr>
</tbody>
</table>

### 1.5.1.3.19 Offline Store Upload Service

The Offline Store Upload API enables a client to upload offline store (database files) to the server. If Offline Store Upload is enabled for an application, you can upload the offline store (database files) into the server. The upload process creates a zip file that includes the uploaded database files, and is then saved into the server database.

Offline Store Upload supports two types of APIs:
- Runtime APIs
- Admin APIs

#### Related Information

- Runtime APIs [page 390]
- Admin APIs [page 392]

### 1.5.1.3.19.1 Runtime APIs

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

#### Table 281: Runtime APIs

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>POST Offline Store Upload [page 391]</td>
</tr>
</tbody>
</table>
1.5.1.3.19.1.1 POST Offline Store Upload

This method allows you to upload the offline store (database files) for an application. If Offline Store Upload is enabled for the specified application, then you can upload the offline store (database files) into the server. A zip file gets created to include the uploaded database files, and gets saved into the server database.

HTTP Method: POST.

URL: https://<HMC base URL>/offlinestoreupload/application/{applicationId}/device/{deviceId}

Request Parameters

Table 282:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>It refers to Application identifier.</td>
<td>String</td>
</tr>
<tr>
<td>deviceId</td>
<td>Required</td>
<td>It refers to Device identifier. Its value gets saved with the uploaded log, if specified.</td>
<td>String</td>
</tr>
<tr>
<td>offlineStore</td>
<td>Required</td>
<td>It is the request body for POST request. It includes multiple files in one upload request. It refers to Database files.</td>
<td>OfflineStore</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Offline Store consists of several databases, which needs to be uploaded in one upload request. Offline Store Upload API adds them into one zip file, and uploads the database files into the server. These uploaded database files are downloaded by administrator/developer together.</td>
<td></td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 283:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Offline Store uploaded.</td>
</tr>
<tr>
<td>400</td>
<td>Bad request. Such as request content is not multipart/form-data, or request does not include two files.</td>
</tr>
<tr>
<td>403</td>
<td>Offline Store upload is not enabled for the application.</td>
</tr>
<tr>
<td>404</td>
<td>No application found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
1.5.1.3.19.2 Admin APIs

The Offline Store Upload API enables a client to upload offline store (database files) to the server, and then developer or administrator can download them using Admin API.

Table 284: Admin APIs

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Offline Store Upload</td>
<td>392</td>
</tr>
<tr>
<td>POST</td>
<td>POST Offline Store Upload</td>
<td>394</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Offline Store Upload</td>
<td>395</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Offline Store Upload</td>
<td>396</td>
</tr>
</tbody>
</table>

1.5.1.3.19.2.1 GET Offline Store Upload

**HTTP Method:** GET

**Service Document**

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/

**Response Status and Error Codes**

Table 285:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Service document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

**Metadata Document**

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/$metadata

**Response Status and Error Codes**

Table 286:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Metadata document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

**Entity set PolicySet**

This method allows you to get the entities from the entity set PolicySet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/PolicySet
Response Status and Error Codes

Table 287:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet PolicySet.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

PolicySet by Key

This method allows you to get the entity from PolicySet by key, and it returns the entity with the key from the PolicySet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/PolicySet('{applicationId}')

Request Parameters

Table 288:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 289:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet PolicySet.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Entity set OfflineStoreSet

This method allows you to get the entities from the entity set OfflineStoreSet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/OfflineStoreSet

Response Status and Error Codes

Table 290:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet PolicySet.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

OfflineStoreSet by Key

This method allows you to get the entity from OfflineStoreSet by key, and returns the entity with the key from OfflineStoreSet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/OfflineStoreSet('{id}')
Request Parameters

Table 291:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Required</td>
<td>key: id</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 292:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>EntitySet OfflineStoreSet.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

OfflineStoreSet value

This method allows you to get the zip file, which includes the two databases for the offline store.

URL: https://<HMC base URL>/Admin/offlinestoreupload/OfflineStoreSet('{id}')/$value

Request Parameters

Table 293:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Required</td>
<td>key: id</td>
<td>String</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 294:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>The binary content of the zip file.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.19.2.2 POST Offline Store Upload

This method allows you to post a new entity to entity set PolicySet.

HTTP Method: POST

URL: https://<HMC base URL>/Admin/offlinestoreupload/PolicySet
Table 295:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
</table>
| Policy    | Yes  | The new entity.  
|           |      | - It is the request body for POST request.  
|           |      | - If its content is in json format, then content-type of request must be \textit{application/json}.  
|           |      | - If its content is xml/atom format, then the request's content-type must be \textit{application/xml}. | • ApplicationId: string  
|           |      |            | • LogUploadEnabled: boolean,null  
|           |      |            | • ExpiryTime: integer,null  
|           |      |            | • MaxFileSize: integer,null |

Response Status and Error Codes

Table 296:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created entity.</td>
</tr>
<tr>
<td>409</td>
<td>Offline Store Upload policy already exists.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.19.2.3 PATCH Offline Store Upload

This method allows you to update the entity in EntitySet PolicySet.

**HTTP Method:** \textit{PATCH}

**URL:** \textit{https://<HMC base URL>/Admin/offlinestoreupload/PolicySet({'applicationId'})}

**Request Parameters**

Table 297:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>String</td>
</tr>
</tbody>
</table>
Parameter | Type | Description | Parameter Type
--- | --- | --- | ---
Policy | Required | The new entity. ● It is the request body for PATCH request. ● If its content is in json format, then content-type of request must be application/json. ● If its content is xml/atom format, then the request’s content-type must be application/xml. | ApplicationId: string LogUploadEnabled: boolean,null ExpiryTime: integer,null MaxFileSize: integer,null

Response Status and Error Codes
Table 298:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.19.2.4 DELETE Offline Store Upload

HTTP Method: **DELETE**

EntitySet PolicySet

This method allows you to delete the entity in EntitySet PolicySet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/PolicySet('{applicationId}')

Request Parameters

Table 299:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 300:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
</tbody>
</table>
EntitySet OfflineStoreSet

This method allows you to delete the entity in EntitySet OfflineStoreSet.

**URL:** https://<HMC base URL>/Admin/offlinestoreupload/OfflineStoreSet('{id}')

### Request Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Required</td>
<td>key: id</td>
<td>String</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

### 1.5.1.3.20 Client Usage Report Upload Service

The Client Usage Report Upload API allows the application to upload client usage reports for SAP Cloud Platform mobile service for development and operations.

Client Usage Report Upload Service supports both Runtime and Admin APIs.

### Related Information

- Runtime APIs [page 398]
- Admin APIs [page 399]
1.5.1.3.20.1 Runtime APIs

This page describes the various operations in the Client Usage Report Upload Runtime APIs.

Table 303: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>POST</td>
<td>POST Client Usage Upload [page 398]</td>
</tr>
</tbody>
</table>

1.5.1.3.20.1.1 POST Client Usage Upload

This method allows clients to upload client usage reports in JSON format.

Example

The upload endpoint allows clients to upload client usage reports in the JSON format.

```json
{
  "report": "reportUUID", (required) "appInfo":
  { "application": "ClientUsageUpload0ae95adbc1e544388b46ab752688b86f", (required) "appVersion": "<Version No.>" }, "deviceInfo":
  { "platform": "<Platform Name>" }, "platformVersion": "<Platform Version No.>" }, "deviceModel": "<Device Model>" }, "deviceID": "<Device ID>" (required) }, "sessions": [], "events": [
  { "type": "type1", "key": "key1", "time": "2014-07-02T12:01:49.178+03", (required) "duration": "5.46760", "screen": "first screen", "view": "view1", "element": "element1", "action": "action1", "behavior": "login", "cases": "case1", "category": "category1", "result": "result1", "unit": "unit1", "measurement": "measurement1", "value": "value1", "others": "other param" }, ]
}
```

HTTP Method: POST

URL: https://<HMC base URL>/mobileservices/application/{applicationId}/clientusage/v1/runtime/clientusage/application/{applicationId}/device/{deviceId}

Table 304:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>reports</td>
<td>Required</td>
<td>- It refers to the uploaded client report file.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- It requires multipart/form-data request header.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- It is supported in JAX-RS.</td>
</tr>
</tbody>
</table>
### Parameter Table

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>• ID of the current application. • It must match the application ID that was used while configuring the client usage upload for your app.</td>
<td>string</td>
</tr>
</tbody>
</table>

### Response Status and Error Codes

Table 305:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>The upload was successful.</td>
</tr>
<tr>
<td>400</td>
<td>The upload failed due to an incorrect payload.</td>
</tr>
<tr>
<td>403</td>
<td>The client usage report upload is not enabled for the given application.</td>
</tr>
<tr>
<td>default</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

### 1.5.1.3.20.2 Admin APIs

This page describes the various operations in Client Usage Report Upload Admin APIs.

Table 306: Admin API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Client Usage Upload [page 399]</td>
</tr>
<tr>
<td>POST</td>
<td>POST Client Usage Upload [page 401]</td>
</tr>
<tr>
<td>PATCH</td>
<td>PATCH Client Usage Upload [page 401]</td>
</tr>
<tr>
<td>DELETE</td>
<td>DELETE Client Usage Upload [page 402]</td>
</tr>
</tbody>
</table>

### 1.5.1.3.20.2.1 GET Client Usage Upload

**HTTP Method:** GET

**Service Document**

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin
Response Status and Error Codes

Table 307:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Service document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Metadata Document

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/$metadata

Response Status and Error Codes

Table 308:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Metadata document.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Entity Set Application Setting

This method allows you to get the entities from the entity set Application Setting.

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/ApplicationSettingSet

Response Status and Error Codes

Table 309:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity Set Application Setting.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

Application Setting by Key

This method allows you to get the entity from the Application Setting by key, and it returns the entity with the key from the Application Setting.

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/ApplicationSettingSet('{applicationId}')

Request Parameters

Table 310:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
</tbody>
</table>
Response Status and Error Codes

Table 311:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Entity Set Application Setting.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.20.2.2 POST Client Usage Upload

This method allows you to post the new entity to the Entity Set Application Setting.

**HTTP Method:** POST

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/ApplicationSettingSet

Table 312:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApplicationSetting</td>
<td>Required</td>
<td>The new entity</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• ApplicationId: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• ClientUsageUploadEnabled: boolean</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 313:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>201</td>
<td>Created entity.</td>
</tr>
<tr>
<td>409</td>
<td>Client usage upload setting already exists..</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.20.2.3 PATCH Client Usage Upload

This method allows you to update the entity in the Entity Set Application Setting.

**HTTP Method:** PATCH

**URL:** https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/ApplicationSettingSet('{applicationId}')

**Request Parameters**
Table 314:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
<tr>
<td>ApplicationSet</td>
<td>Required</td>
<td>The entity to patch.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● ApplicationId: string</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>● ClientUsageUploadEnabled: boolean</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 315:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not Found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>

1.5.1.3.20.2.4 DELETE Client Usage Upload

This method allows you to delete an entity in the Entity Set Application Setting.

HTTP Method: DELETE

URL: https://<HMC base URL>/mobileservices/origin/{origin}/clientusage/v1/admin/ApplicationSettingSet('{applicationId}')

Request Parameters

Table 316:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>applicationId</td>
<td>Required</td>
<td>key: applicationId</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 317:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>204</td>
<td>Empty response.</td>
</tr>
<tr>
<td>404</td>
<td>Not found.</td>
</tr>
<tr>
<td>500</td>
<td>Unexpected error.</td>
</tr>
</tbody>
</table>
1.5.1.3.21 Role Service

This service allows you to get the logical roles that are assigned to the current user, which you can use to build flexible UIs for a particular mobile application based on the roles that are assigned to the user.

Prerequisite: Access Control Policy

In the access control policy, define which roles are relevant for the current mobile application. The Role service checks each role in the Access Control policy, and returns those that are assigned to the current user.

Note

If the user does not have the required role and tries to register for an application, the access control policy returns a 403 error message.

The Role service supports Runtime APIs.

Related Information

Runtime APIs [page 403]

1.5.1.3.21.1 Runtime APIs

The Role service API allows the application to retrieve user and roles for the currently authorized user, using the SCIM protocol.

Table 318: Runtime API

<table>
<thead>
<tr>
<th>HTTP Method</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>GET Role Service [page 403]</td>
</tr>
</tbody>
</table>

1.5.1.3.21.1.1 GET Role Service

Gets all the logical roles belonging to the current user.

HTTP Method: GET

URL: https://<HMC base URL>/mobileservices/application/{appId}/roleservice/application/{appId}/v2/Me
Request Parameters

Table 319:

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Description</th>
<th>Parameter Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>appId</td>
<td>Required</td>
<td>The current application to use the role service.</td>
<td>string</td>
</tr>
</tbody>
</table>

Response Status and Error Codes

Table 320:

<table>
<thead>
<tr>
<th>Code</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>200</td>
<td>Retrieval of the roles associated with the user is successful.</td>
</tr>
<tr>
<td>403</td>
<td>The user does not have the required authorization to access this application.</td>
</tr>
<tr>
<td>404</td>
<td>Application ID does not exist.</td>
</tr>
<tr>
<td>412</td>
<td>The Application does not enable access control.</td>
</tr>
</tbody>
</table>

1.5.1.4 Application Connection Properties

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

**Note**

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

Onboarding Version 1 or Later

Table 321: Application Connection Properties: Uncategorized

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ETag</td>
<td>Specifies the version identifier.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ApplicationConnectionId</td>
<td>ID that uniquely identifies an application. Usually generated by the server and in the format of a GUID.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
### Table 322: Application Connection Properties: Android Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>AndroidGcmPushEnabled</td>
<td>Indicates if Google Cloud Messaging (GCM) push notifications are enabled and configured for this application.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>AndroidGcmRegistrationId</td>
<td>The registration ID that the device acquires from Google during GCM registration.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>AndroidGcmSenderId</td>
<td>GCM sender ID used by SAP Mobile Server to send notifications. Used by the client to register for GCM.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Table 323: Application Connection Properties: Apple Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ApnsPushEnable</td>
<td>Indicates if push notification using APNs is enabled or not.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>ApnsDeviceToken</td>
<td>The Apple push notification service token.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Table 324: Application Connection Properties: Application Settings

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomizationBundleId</td>
<td>The application configuration (customization resource bundles) associated with the application.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ApplicationVersion</td>
<td>The version number of the registered application.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>ClientSdkVersion</td>
<td>The version number of the SDK for the registered application.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Table 325: Application Connection Properties: BlackBerry Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>BlackBerryPushEnabled</td>
<td>Indicates if BlackBerry push notifications are enabled and configured for this application.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Property Name</td>
<td>Description</td>
<td>Type</td>
<td>Read-only?</td>
<td>Is Nullable?</td>
</tr>
<tr>
<td>-----------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>BlackberryDevicePin</td>
<td>Every Blackberry device has a unique permanent PIN. During initial connection and settings exchange, the device sends this information to the server.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>BlackberryBESListenerPort</td>
<td>The listener port for BES notifications.</td>
<td>Int32</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 326: Application Connection Properties: Windows Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>WnsChannelURI</td>
<td>The WNS Channel URI as provided by WNS during push registration.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>WnsPushEnable</td>
<td>Indicates if push notification using Wns is enabled or not.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 327: Application Connection Properties: MPNS Push

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MpnsChannelURI</td>
<td>The MPNS Channel URI as provided by MPNS during push registration.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>MpnsPushEnable</td>
<td>Indicates if push notification using Mpns is enabled or not.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 328: Application Connection Properties: Capabilities

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CapabilitiesPasswordPolicy</td>
<td>Updates the password policy for an application connection.</td>
<td>Boolean</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 329: Application Connection Properties: Custom Settings

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>CustomCustom1</td>
<td>Available for free use.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom2</td>
<td>Available for free use.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom3</td>
<td>Available for free use.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CustomCustom4</td>
<td>Available for free use.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Table 330: Application Connection Properties: Device Information

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeviceModel</td>
<td>The manufacturer of the registered mobile device.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceType</td>
<td>The type of device. Supported device types are: WinMobile, WinSmartPhone, Windows, iPhone, iPad, iPod, iOS, BlackBerry, Android, Rim6, Windows8, WinPhone8, Windows81, WinPhone81.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceSubType</td>
<td>The device subtype of the device.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DevicePhoneNumber</td>
<td>The phone number associated with the registered mobile device. This phone number is used when sending SMS text messages to a device. The phone number must be provided in international format, e.g. “+491712345678” or “+15551231234”.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>DeviceIMSI</td>
<td>The International Mobile Subscriber identity, which is a unique number associated with all Global System for Mobile communication (GSM) and Universal Mobile Telecommunications System (UMTS) network mobile phone users.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Table 331: Application Connection Properties: Password Policy

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>PasswordPolicyEnabled</td>
<td>Specifies whether password policies are enabled.</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyDefaultPasswordAllowed</td>
<td>Specifies whether default passwords are allowed.</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Property Name</td>
<td>Description</td>
<td>Type</td>
<td>Read-only?</td>
<td>Is Nullable?</td>
</tr>
<tr>
<td>-------------------------------</td>
<td>------------------------------------------------------------------------------</td>
<td>----------</td>
<td>------------</td>
<td>--------------</td>
</tr>
<tr>
<td>PasswordPolicyMinLength</td>
<td>Specifies how long the password chosen by the user must be.</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyDigitRequired</td>
<td>Specifies whether the password must contain digit(s).</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyUpperCase</td>
<td>Specifies whether the password must contain uppercase characters.</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyLowerCase</td>
<td>Specifies whether the password must contain lowercase characters.</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicySpecialRequired</td>
<td>Specifies whether the password must contain non-alphanumeric characters.</td>
<td>Boolean</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyExpireInNDays</td>
<td>Specifies the number of days the existing password can be used before it must be changed by the user.</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyMinUniqueChars</td>
<td>Determines how many unique characters must be used in the password.</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyLockTimeout</td>
<td>Determines how long a successfully unlocked data vault will remain open. When the timeout expires, the vault is locked, and the user must re-enter the vault password to resume using the application.</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PasswordPolicyRetryLimit</td>
<td>Determines how long a successfully unlocked data vault will remain open. When the timeout expires, the vault is locked, and the user must re-enter the vault password to resume using the application.</td>
<td>Int32</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>
Table 332: Application Connection Properties: Proxy

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>ProxyApplicationEndpoint</td>
<td>The URL pointing to the EIS.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ProxyPushEndpoint</td>
<td>The SAP Mobile Platform URL for sending out notifications.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 333: Application Connection Properties: Usage

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxConnectionWaitTimeForClientUsage</td>
<td>Determines how long a connection exists for client usage.</td>
<td>Int32</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>EnableAppSpecificClientUsageKeys</td>
<td>Determines if the application developer can use custom information in Usage Data Collection.</td>
<td>Boolean</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 334: Application Connection Properties: Log

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UploadLogs</td>
<td>Specifies whether log upload is enabled for this app and device registration.</td>
<td>Boolean</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>LogEntryExpiry</td>
<td>Specifies the maximum time the logs would be kept on the device before they get removed.</td>
<td>Int32</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 335: Application Connection Properties

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>E2ETraceLevel</td>
<td>The log level to be used for End 2 End traces.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>PublishedToMobilePlace</td>
<td>Determines if the configuration is currently shared through mobile place or not.</td>
<td>Boolean</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>FeatureVectorPolicyAllEnabled</td>
<td>Determines if the hybrid capability policy is enabled or not. If enabled then the administrator can select which features to allow to work per application. If disabled all features will work.</td>
<td>Boolean</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>
### Onboarding Version 3 or Later

Table 336: Application Connection Properties: Capability

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Category</td>
<td>The capability category.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>CapabilityName</td>
<td>Name of the capability.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>ApplicationConnectio­nId</td>
<td>ID that uniquely identifies an application.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>CapabilityValue</td>
<td>Value of the capability.</td>
<td>String</td>
<td>Yes</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 337: Application Connection Property: Form

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>FormFactor</td>
<td>The form factor of the device. Mostly used in combination with Capability based push.</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

### Onboarding Version 4 or Later

Table 338: Application Connection Property: UserName

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserName</td>
<td>Specifies the user name that was used during registration</td>
<td>String</td>
<td>Yes</td>
<td>No</td>
</tr>
</tbody>
</table>

Table 339: Application Connection Property: Application Settings

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Description</th>
<th>Type</th>
<th>Read-only?</th>
<th>Is Nullable?</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserLocale</td>
<td>Specifies the user preferred language.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TimeZone</td>
<td>Specifies the time zone.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>LastKnownLocation</td>
<td>Specifies last known location of the app sent to the server.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CreatedAt</td>
<td>Indicates the time the application was created.</td>
<td>DateTime</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>PushGroup</td>
<td>Custom field to specify a target audience for push information.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Email</td>
<td>Users email address.</td>
<td>String</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>
1.6 Migration

With minimal disruption, you can manually migrate applications that are running on SAP Mobile Platform to SAP Cloud Platform mobile service for development and operations.

Migration Scenario

A completely seamless migration is impossible, due to the environmental differences between SAP Mobile Platform and SAP Cloud Platform mobile service for development and operations. A migration affects both the local architecture and the user experience. The following table describes the source and corresponding target landscape for an SAP Mobile Platform application.

<table>
<thead>
<tr>
<th>Pre-Migration Landscape</th>
<th>Post-Migration Landscape (SAP Cloud Platform mobile service for development and operations)</th>
</tr>
</thead>
<tbody>
<tr>
<td>SAP Mobile Platform 3.x installed</td>
<td>SAP Cloud Platform mobile service for development and operations installed</td>
</tr>
<tr>
<td>SAP NetWeaver Gateway provides OData services for the mobile application to be migrated (on-premise)</td>
<td>SAP NetWeaver Gateway provides OData services for the mobile applications to be migrated (on-premise)</td>
</tr>
<tr>
<td>Mobile applications are hybrid Android, iOS, or Windows 8.1 applications, which use the Mobile Application Framework (MAF) Logon plug-in.</td>
<td>Mobile applications are hybrid Android, iOS, or Windows 8.1 applications, which use the Mobile Application Framework (MAF) Logon plug-in.</td>
</tr>
<tr>
<td>Mobile user authentication uses basic HTTP against the SAP NetWeaver Gateway system on-premise and external OData services.</td>
<td>Mobile user authentication uses basic HTTP against the SAP NetWeaver Gateway system on-premise and external OData services.</td>
</tr>
</tbody>
</table>

Applications You Cannot Migrate

Migration is not possible for:

- Agentry-based applications
- Mobiliser-based applications
- Mobile applications that require custom OSGi bundles; in this scenario, you must migrate the code-base bundle to SAP Cloud Platform mobile service for app and device management.
- Applications that are based on mobile business object (MBO) technology
- Applications that use customOSGi authentication modules
- Short Message Service (SMS) based applications
1.6.1 Migrating to SAP Cloud Platform Mobile Service for Development and Operations

Manually migrate an SAP Mobile Platform application to the mobile service for development and operations.

Procedure

1. Collect the required information for migrating from the source system:
   - Back-end service URL
   - Back-end authentication service URL
   - Application ID
   - URL rewrite options
   - Proxy type
   - Authentication option
   - Client password policy
   - Client log policy
   - Push settings
   - Client resources
   - Application-specific settings
2. Install SAP Cloud Connector. See Installing the Cloud Connector.
3. On the Access Control in SAP cloud connector, whitelist the necessary back-end service URLs.
   - To use an internal OData URL, select OnPremise as the proxy type.
   - To use an Internet OData Service URL, select Internet as the proxy type.
   See Configuring the Cloud Connector for HTTP
4. In SAP Cloud Platform mobile service for development and operations, create an application. See Configuring Applications.
5. (Optional if you use your mobile app for testing) Test the application configuration with REST client. See REST API Application Development Overview.
   Ensure that the initial server configuration is complete and working.
6. Migrate the client application to the latest SAP Mobile SDK. See SAP Mobile Platform SDK in SAP Mobile Platform.
7. Check the network connectivity settings of the device.
8. Use corporate WiFi to connect to SAP Mobile Platform.
9. Configure and enable WiFi with Internet access to connect to SAP Cloud Platform mobile service for development and operations.
10. Before onboarding the client application on the device, verify that Mobile Service for Development and Operations cockpit is reachable.
11. For applications with offline features, check the proxy setting in the client application source code.
Note
For example, if the offline store cannot open or has a network error during runtime for ODataOfflineStoreOption object in the following format:
MyODataOfflineStoreOptions.extraStreamParms="proxy_host=myproxy;proxy_port=8080";

12. Run a complete regression test.

Note
Already enrolled apps will not automatically be registered with SAP Cloud Platform mobile service for development and operations; you must reregister these apps with the new URL

Related Information
Configuring Applications [page 43]

1.7 Glossary

Defines terms and components for SAP Cloud Platform mobile service for development and operations.

- **anonymous user**: A user who can access the system without providing identification.
- **Apple Push Notification Service (APNS)**: A free service provided by Apple for devices running iOS. The APNS pushes notifications from a provider to a device, which means applications need not operate as active listeners for those notifications.
- **application user**: A distinct set of identities (identified or anonymous) that are in contact with the system by using an application. In Mobile Services Cockpit, an application user is the distinct list of names under which a user has been identified to the system. An application user may also be a user (identified or anonymous) that has been associated with an application ID.
- **back end**: A system that provides a data source, such as a database or Web service.
- **certificate**: A digital security mechanism attached to an electronic message that verifies the identity of a specific user.
- **client application**: In SAP Cloud Platform mobile service for development and operations, the software that runs on a smart phone, tablet computer, or other mobile device. See mobile application.
- **client resources**: Also known as resource bundles. Containers used by applications to download dynamic configurations, styles, or content from the cloud.
- **connection**: Configuration details and credentials that are required to connect to a database, Web service, or other back end.
<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>data vault</td>
<td>Provides encrypted storage of occasionally used, small pieces of data from multiple operational systems.</td>
</tr>
<tr>
<td>deploy</td>
<td>To upload a computer program or development unit from a development state to a server, moving it from a packaged or assembled form to an operational working state that can be consumed. SAP Cloud Platform mobile service for development and operations can then make the unit accessible to users via a client application that is installed on a mobile device.</td>
</tr>
<tr>
<td>device application</td>
<td>A software application that runs on a mobile device. See mobile application.</td>
</tr>
<tr>
<td>discovery service</td>
<td>Provides the configuration information necessary for a user to enroll a device with SAP Mobile Secure. Allows you to distribute initial configuration data to mobile apps to enhance the user onboarding process.</td>
</tr>
<tr>
<td>export</td>
<td>The movement of mobile objects from a system so they can be imported into another system. Typically performed by the SAP Cloud Platform mobile services administrator.</td>
</tr>
<tr>
<td>Google Cloud Messaging (GCM)</td>
<td>A free service offered by Google for sending messages to Android devices. Requires an API key to allow SAP Cloud Platform mobile service for development and operations to send push notifications over GCM.</td>
</tr>
<tr>
<td>hybrid application</td>
<td>An application developed using Web technologies, such as HTML5 and JavaScript, that runs within a native application on the device. The container provides the Web application with access to native device capabilities through an exposed JavaScript API.</td>
</tr>
<tr>
<td>keystore</td>
<td>The location in which encryption keys, digital certificates, and other credentials in either encrypted or unencrypted keystore file types are stored for SAP Cloud Platform mobile services runtime components. See truststore.</td>
</tr>
<tr>
<td>mobile application (mobile app)</td>
<td>A software application designed to run on smart phones, tablet computers, and other mobile devices.</td>
</tr>
<tr>
<td>Mobile Services Cockpit</td>
<td>A Web-based interface in SAP Cloud Platform mobile service for development and operations for creating and administering mobile applications, registering users, creating and maintaining connections, and performing administration tasks related to reporting, logging, and onboarding.</td>
</tr>
<tr>
<td>monitoring</td>
<td>A SAP Cloud Platform mobile service for development and operations feature that allows administrators to identify areas of weakness or periods of high activity in a particular area, as well as overall system health. Use for system diagnostics or for troubleshooting.</td>
</tr>
<tr>
<td>Microsoft Push Notification Service for Windows Phone (MPNS)</td>
<td>A free service that enables you to send push notification messages to Windows Phone 7+ and Windows Phone 8.0 apps.</td>
</tr>
<tr>
<td>Open Data Protocol (OData)</td>
<td>Provides standard create, read, update, and delete (CRUD) access to a data source via a web site. OData is similar to JDBC and ODBC, although not limited to SQL databases.</td>
</tr>
<tr>
<td><strong>OData proxy</strong></td>
<td>A connection to the mobile server that funnels OData service requests through the platform, giving administrators and developers more control by forcing only whitelisted endpoints to be accessible from the application. Also restricts who is able to access the endpoint, based on security mechanisms that are built into the platform.</td>
</tr>
<tr>
<td><strong>onboarding</strong></td>
<td>The enterprise-level activation of an authentic device, a user, and an application entity as a combination in SAP Cloud Platform mobile service for development and operations.</td>
</tr>
<tr>
<td><strong>Security Assertion Markup Language (SAML)</strong></td>
<td>An XML-based open standard data format for exchanging authentication and authorization data between an identity provider and a service provider.</td>
</tr>
<tr>
<td><strong>SAP Fiori</strong></td>
<td>The user experience (UX) for SAP software; represents a personalized, responsive, and simple user experience across devices and deployment options.</td>
</tr>
<tr>
<td><strong>SAP Cloud Platform</strong></td>
<td>Platform as a Service (PaaS) offering from SAP; enables customers and developers to build, extend, and run applications on SAP in the cloud.</td>
</tr>
<tr>
<td><strong>SAP Mobile Place</strong></td>
<td>An SAP mobile application management offering that is a brandable, localizable, and secure enterprise app store, making it easy for companies to push their mobile apps to employees, business partners, and consumers.</td>
</tr>
<tr>
<td><strong>SAP Mobile Secure</strong></td>
<td>A cloud-based SAP Enterprise Mobility Management (EMM) offering.</td>
</tr>
<tr>
<td><strong>schedule</strong></td>
<td>The definition of a task (such as the collection of a set of statistics) and the time interval during which the task must execute using SAP Cloud Platform mobile services.</td>
</tr>
<tr>
<td><strong>System for Cross-domain Identity Management (SCIM)</strong></td>
<td>An open standard that connects SAP Cloud Platform mobile service for development and operations to a back-end authentication user store.</td>
</tr>
<tr>
<td><strong>security configuration</strong></td>
<td>The mechanism within SAP Cloud Platform that enforces application authentication and authorization. The security configuration points the platform to an underlying user store (a repository, such as Active Directory or an LDAP server) to perform authentication and authorization services.</td>
</tr>
<tr>
<td><strong>single sign-on (SSO)</strong></td>
<td>A credential-based authentication mechanism for accessing multiple, but independent, software systems using a single logon.</td>
</tr>
<tr>
<td><strong>truststore</strong></td>
<td>The location in which certificate authority (CA) signing certificates are stored. See keystore.</td>
</tr>
<tr>
<td><strong>Windows Push Notification Service (WNS)</strong></td>
<td>A free service that enables third-party developers to send toast, tile, badge, and raw updates from their own cloud service to Windows Store apps. All modern UI apps can receive notifications via WNS, but not traditional desktop applications. See MPNS for information about push notification service to Windows Phone.</td>
</tr>
</tbody>
</table>
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).