



Using the OpenEdge Mobile Push Notification API for Kinvey

Overview

A push notification is a message that is sent by a mobile app publisher to a mobile app user's device. The message is displayed either in the notification pane or in the mobile app itself.

The Progress OpenEdge Mobile Push Notification API enables you to send push notifications from an ABL application to supported mobile devices through Kinvey.

Kinvey is a Backend As A Service (BaaS) platform that provides a rich set of features on the cloud such as user management, data storage, and push notifications, that take away a lot of the complexity involved in developing a mobile app.

Supported platforms and features

Currently, the OpenEdge Mobile Push Notification API supports sending push notifications to the following mobile operating systems:

- Android
- iOS

For this release, you can only broadcast a push notification to all devices. We are working on adding support for filtering and scheduling, which we plan to ship in the near future.

Using Kinvey instead of Telerik Platform

The OpenEdge Mobile Push Notification API delivers push notification messages to Kinvey, which in turn transports the messages to a push notification server for Android or iOS.

The Telerik Platform has been retired and the OpenEdge Telerik Push Notification API is now deprecated. Although, the OpenEdge Mobile API still supports Telerik Platform, if you are a Telerik Platform user, you must migrate your application to Kinvey and modify your ABL application code to use the new OpenEdge Mobile Push Notification API.

Note: At this point in time, there is no substitute for the **TelerikPushNotificationAdmin** API. Admin functions for Kinvey, such as those encapsulated by the **TelerikPushNotificationAdmin** API, are not supported in this release.

You may find the following resources useful if you are migrating from Telerik Platform to Kinvey:

- [Telerik Platform Migration Guide](#)
- [Push notification differences between Telerik Platform and Kinvey](#)

Architecture

Unlike Telerik Platform, Kinvey does not have a dedicated endpoint to receive push notifications. Instead, you must create a custom endpoint in Kinvey and use the OpenEdge Mobile Push Notification API to send JSON messages to that endpoint.

The custom endpoint is a piece of JavaScript code that you must write to handle incoming messages from OpenEdge, form a push notification payload, and send the payload to notification servers (Google Cloud Messaging for Android and Apple Push Notification Service for iOS). The notification servers, in turn, deliver the push notifications to mobile devices.

Mobile apps must use the Kinvey client library for Android or iOS to be able to handle and display the push notifications from Kinvey. They also need to be integrated with the push notification server used by their operating system.

The task of registering and managing users and device tokens used for push notifications is performed by Kinvey.



Key API classes

The OpenEdge Mobile Push Notification API comprises the following classes:

Class	Description
<code>OpenEdge.Mobile.PushNotificationService</code>	You must instantiate a Push Notification service. This class provides methods to send push notification messages to a custom endpoint in Kinvey.
<code>OpenEdge.Mobile.PushNotificationMessageBuilder</code>	Provides methods to build a push notification message that can be passed to the Push Notification service. This is optional. You can also choose to pass a JSON object to the Push Notification service.

Using the OpenEdge Mobile Push Notification API to send push notifications to Kinvey

Perform the following tasks to send push notifications to Kinvey from an ABL application:

1. **Download the updated OpenEdge .Net procedure library.** (You can skip this step if you are using OpenEdge 11.7.4).

The `OpenEdge.Net` procedure library has been updated in OpenEdge 11.7.4 to include the Mobile Push Notification API for Kinvey. If your OpenEdge version is older than 11.7.4, [download](#) the updated procedure library. Copy this updated library to the following locations, replacing the existing `.pl` files:

- `<OpenEdge_install_directory>\gui\netlib`
- `<OpenEdge_install_directory>\tty\netlib`
- `<OpenEdge_install_directory>\src\netlib`

Ensure that the **PROPATH** settings for your OpenEdge project point to the updated `OpenEdge.Net` procedure library.

2. **Download and install the root CA certificate used to initiate an SSL handshake with Kinvey.**

Your OpenEdge application must establish an SSL connection with Kinvey before it can send push notification messages. To enable the SSL handshake, you need to [download](#) the root SSL certificate and install it in the OpenEdge certification store as follows.

Note: The root certificate authority (CA) is Comodo RSA. Kinvey's public SSL certificate is derived from this CA.

Launch **Proenv** and run the `certutil` utility:

```
certutil -import certificate-filepath
```

3. Import the following API classes into the ABL class from which you want to send push notifications.

```
using OpenEdge.Mobile.PushNotificationService.
using OpenEdge.Mobile.PushNotificationMessageBuilder.
```

4. Create a push notification message payload.

You have two options here--you can either create a JSON object or a `PushNotificationMessageBuilder` object.

- **Creating a JSON object**

1. Create a JSON file containing a JSON message with the fields that are shown [in this example](#).
2. In the ABL class, import the following API classes:

```
using Progress.Json.ObjectModel.ObjectModelParser.
using Progress.Json.ObjectModel.JsonObject.
```

3. Build the JSON object as shown in this example:

```
define variable oPayload as JsonObject no-undo.
define variable oParser as ObjectModelParser no-undo.

oPayload = new JsonObject().
oParser = new ObjectModelParser().
oPayload = cast(oParser:ParseFile(search('MyJsonFile.json')), JsonObject).
```

- **Creating a `PushNotificationMessageBuilder` object**

1. Create a String message:

```
define variable cMessage as character no-undo.
cMessage = 'Payload generated using builder'.
```

2. Create a badge (required for iOS only):

```
define variable iBadge as integer no-undo.
iBadge = 0.
```

3. Build the `PushNotificationBuilder` object as shown in this example:

```
define variable oBuilder as PushNotificationMessageBuilder no-undo.
oBuilder = PushNotificationMessageBuilder:Send(cMessage, iBadge)
```

5. Set up a custom endpoint in Kinvey.

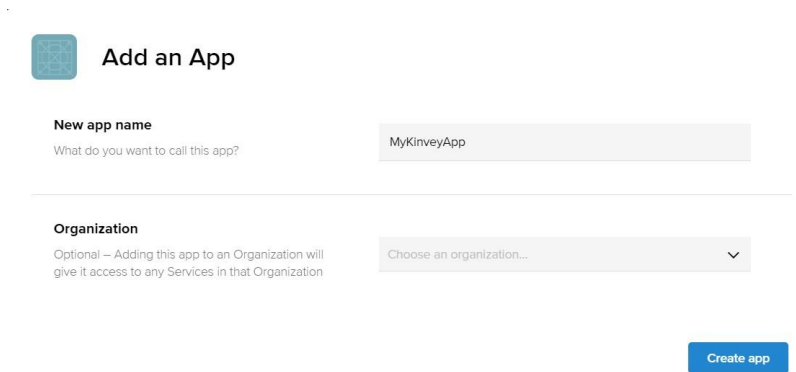
Before you can send push notification messages to Kinvey, you need to set up a custom endpoint in a Kinvey app. Kinvey provides a cloud-based console that you can open in a browser. You use this console to create a **Kinvey app**, which gives you access to all of Kinvey's cloud-based services, including the ability to create custom endpoints.

A **custom endpoint** is a URI to which you can send HTTP requests. You can write custom endpoint code in JavaScript that gets executed when a request is received by the endpoint. For push notifications, you need to write JavaScript code that processes HTTP requests from your ABL application and uses Kinvey APIs to send push notifications to notification servers. These notification servers must be configured separately in the Kinvey app.

Perform the following steps to set up a custom endpoint in Kinvey:

1. [Sign up](#) for a Kinvey account. Note that your account gets associated with a free **Developer** plan by default.
2. [Log into](#) the Kinvey console.

3. Create a Kinvey app by clicking **Add an app** and then following the instructions in the dialog box that opens.



4. Create a custom endpoint as follows:

1. Click **Custom Endpoints** under **Business Logic**.
2. Click **+Add an Endpoint**.
3. In the dialog box that opens, enter a name for the endpoint, select **Code editor** and then click **Create an endpoint**.
4. Write endpoint code as shown in [this example](#). The Kinvey custom endpoint specifies an `onRequest()` method. Requests to the endpoint are passed to this method via the `request` parameter. The `modules` parameter loads Kinvey's modules, which contain the APIs that you need to process the request. For push notifications, you need to load two APIs--`push` (for pushing messages to notification servers) and `collectionAccess` (for accessing user data).

6. Use the OpenEdge push notification service to send the push notification payload.

To send a push notification message from an ABL application to a Kinvey custom endpoint, you need to have the following information:

- The custom endpoint name.
- Valid username and password credentials to access the Kinvey app.
- The Kinvey app's **App Key**.

Note: You can find the App Key by opening the home page of your Kinvey app and clicking the ellipsis next to the app name.

Perform the following steps to send the push notification payload:

1. Create a `Credentials` object to store the Kinvey app credentials:

```
define variable oCredentials as Credentials no-undo.
define variable cUsername as character no-undo.
define variable cPassword as character no-undo.

cUsername = 'myUsername'.
cPassword = 'myPassword'.

oCredentials = new Credentials('baas.kinvey.com', cUsername, cPassword).
```

Note: The `Credentials` class is part of the `OpenEdge.Net.HTTP` library.

2. Instantiate the Push Notification service as a Kinvey Push Notification Service, passing the Kinvey App Key, custom endpoint name, and the `Credentials` object as parameters:

```
define variable oPushService as PushNotificationService no-undo.  
define variable cAppKey as character no-undo.  
define variable cKinveyCustomEndpoint as character no-undo.  
  
cAppKey = 'kid_Example'.  
cKinveyCustomEndpoint = 'myKinveyCustomEndpoint'.  
  
oPushService = new KinveyPushNotificationService(cAppKey, cKinveyCustomEndpoint,  
oCredentials).
```

3. Initialize the Push Notification service:

```
oPushService:Initialize().
```

4. Send the push notification message by passing the payload object to the `SendNotification()` method. The payload object can be a JSON object or a `PushNotificationMessageBuilder` object.

```
oPushService:SendNotification(oPayload).
```

Additional documentation resources

To learn more about push notifications in Kinvey, see the [Kinvey Android](#) or the [Kinvey iOS](#) documentation.

Push notification differences between Kinvey and Telerik Platform

The push notification mechanism in Kinvey addresses a broad range of requirements. However, there are a few differences between Kinvey and Telerik Platform that you may want to keep in mind when you are migrating your app from Telerik Platform to Kinvey:

- **Amazon SNS for iOS push notifications**—Kinvey uses Amazon SNS under the covers to communicate with Apple Push Notification Service.
- **Google Cloud Messaging (GCM) for Android push notifications**—Kinvey uses GCM to send Android push notifications.
- **Targeted notifications**—To send targeted notifications in Kinvey (for example by filtering for a group of users/devices), you must manually set system properties such as device model, OS version, etc.
- **Windows support**—Windows and Windows Phone are not supported in Kinvey.
- **Pushing messages from client**—In Kinvey, push notifications can only be sent through business logic defined in a custom endpoint or through the Kinvey Console (Web UI). Security policies are not present out of the box; you need to implement them in the business logic.
- **User-based registrations**—In Kinvey, registrations for push notifications are based on the user, not on the device. For this reason, Kinvey does not provide a UI-based browser for registered devices. However, you can view registered users in the Kinvey Console.
- **Scheduling push notifications**—Currently, Kinvey does not provide the option to schedule push notifications.
- **Debugging push notifications**—Kinvey does not provide debugging capabilities to identify failed push notifications.