


Tag and Broadcast Notifications in Native Android Applications

Relevant to:

 Native Android

Overview

Prerequisite: Make sure that you read the [Push notifications in native Android applications](#) tutorial first.

Tag notifications are notification messages that are targeted to all the devices that are subscribed to a particular tag.

Tags represent topics of interest to the user and provide the ability to receive notifications according to the chosen interest.

Broadcast notifications are a form of tag push notifications that are targeted to all subscribed devices. Broadcast notifications are enabled by default for any push-enabled MobileFirst application by a subscription to a reserved `Push.all` tag (auto-created for every device). Broadcast notifications can be disabled by unsubscribing from the reserved `Push.all` tag.

Agenda

- [Notifications configuration](#)
- [Notifications API](#)

Notifications configuration

Tag Notifications configuration

Setting up tags

Tags are defined in the `application-descriptor.xml` file:

```
<nativeAndroidApp xmlns="http://www.worklight.com/native-android-descriptor"
  id="NativeAndroidTagNotifications" platformVersion="7.0.0.00.20150312-0731"
  version="1.0">

  <pushSender key="API_KEY" senderId="PROJECT_NUMBER"/>
  ...
  ...
  ...
  <tags>
    <tag>
      <name>my tag 1</name>
      <description>About my tag 1</description>
    </tag>
    <tag>
      <name>my tag 2</name>
      <description>About my tag 2</description>
    </tag>
  </tags>
</nativeAndroidApp>
```

Notifications API

API methods for tag notifications

Client-side API

- `WLPush.subscribeTag(tagName, options)` – Subscribes the device to the specified tag name
- `WLPush.unsubscribeTag(tagName, options)` – Unsubscribes the device from the specified tag name
- `WLPush.isTagSubscribed(tagName)` – Returns whether the device is subscribed to a specified tag name

Common API methods for tag and broadcast notifications

Client-side API

- `WLNotificationListener`
Defines the callback method to be notified when the notification arrives.
- `client.getPush().setWLNotificationListener(listener)`
This method sets the implementation class of the `WLNotificationListener` interface.
- `client.getPush().setOnReadyToSubscribeListener(listener)`
This method registers a listener to be used for push notifications. This listener should implement the `onReadyToSubscribe()` method.
- The `onMessage(props, payload)` method of `WLNotificationListener` is called when a push notification is received by the device.
 - **props** – A JSON block that contains the notifications properties of the platform.
 - **payload** – A JSON block that contains other data that is sent from MobileFirst Server. The JSON block also contains the tag name for tag-based or broadcast notification. The tag name appears in the “tag” element. For broadcast notification, the default tag name is `Push.ALL`.

Server-side API

`WL.Server.sendMessage(applicationId, notificationOptions)`

This method submits a notification that is based on the specified target parameters.

- `applicationId` – (mandatory) The name of the MobileFirst application
- `notificationOptions` – (mandatory) A JSON block containing message properties

For a full list of message properties, refer to the `WL.Server.sendMessage` API in the API reference of user documentation

Sample application

Before running the application, check the adapter’s `PushAdapter-impl.js` file and verify that the `WL.Server.sendMessage()` method use the correct application name. The correct application name can be determined from the `id` attribute in `application-descriptor.xml`.

[Click to download](#) the MobileFirst project.

[Click to download](#) the Native project.

- The `TagNotifications` project contains a MobileFirst native API that you can deploy to your MobileFirst Server instance.
- The `TagNotificationsAndroid` project contains a native android application that uses a MobileFirst native API library to subscribe to push notifications and receive notifications from GCM.
- Make sure to update the `wlclient.properties` file in the native project with the relevant server settings.