Push Notifications Overview

Overview

IBM MobileFirst Platform Foundation provides a unified set of API methods to send notifications to iOS, Android, Windows 8 Universal, Windows 10 UWP and Cordova (iOS, Android) applications.

This tutorial provides an introduction to push notifications and the supported notifications types, required setup steps to ready the MobileFirst Server to be able to send notificaitons, and the setup steps to ready Native and Cordova applications with support for push notifications, as well as delving into supported scenarios such as sending notifications to applications with and without authentication and available notification types.

Prerequisites:

- MobileFirst Server to run locally, or a remotely running MobileFirst Server.

Jump to:

What is Push Notification?

Push notifications is the ability of a mobile device to receive messages that are "pushed" from a server. Notifications are received regardless of whether the application is currently running in the foreground or background.

Notifications can take several forms:

- **Alert (all)** - a pop-up text message
- **Badge (iOS), Tile (Windows 8.1 Universal, Windows 10 UWP)** - a graphical representation that allows a short text or image
- **Banner (iOS), Toast (Windows 8.1 Universal, Windows 10 UWP)** - a disappearing pop-up text message at the top of the device display
- **Sound (all)** - a sound file playing when a notification is received
- **Interactive (iOS 8 and above)** - action buttons inside the banner of a received notification
- **Silent (iOS 7 and above)** - sending notifications without distrubing the user

Device support

Push notifications are supported for the following mobile platforms:

- Android 2.3.5, 4.x, 5.x, 6.x
- iOS 6, 7, 8 and 9
- Windows 8.1 Universal
- Windows 10 UWP

Push Notification Types

Tag notifications

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

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.

User Authenticated Notifications

User Authenticated Notifications are notifications secured with OAuth.

> For more information about notifications types, see the topic about push notifications in the user documentation.

Setting up support for Push Notifications

Provide the steps to ready the server with push notifications support for iOS and Android (Windows is post-beta(?)).

This section assumes you have an application registered in the server, and should explain how to add a certificate for iOS (as well mention what are the available certificate types) and where to get it from, how to create GCM credentials and how to add them.

In the MFP Console > Security > Map Scope Elements to Security Checks > add the scope *push.mobileclient* to enable push notifications.

Android

[screenshot] In the push services > push settings section of your application in the MFP operations console, add your Google Cloud Messaging (GCM) Push credentials.

To enable GCM for your application, set it up at this link (https://developers.google.com/mobile/add?platform=android&cntapi=gcm&cnturl=https:%2F%2Fdevelopers.google.com%2Fcloud-messaging%2Fandroid%2Fclient&cntlbl=Continue%20Adding%20GCM%20Support&%3Fconfigured%3Dtrue). Keep the configuration file for later. Enter in the Sender ID and API Key into the Push > Push Settings > GCM Push Credential Sections.

Android Push Notifications Service

If your organization has a firewall that restricts the traffic to or from the Internet, you must go through the following steps:

Configure the firewall to allow connectivity with GCM in order for your GCM client apps to receive messages. The ports to open are 5228, 5229, and 5230. GCM typically uses only 5228, but it sometimes uses 5229 and 5230. GCM does not provide specific IP, so you must allow your firewall to accept outgoing connections to all IP addresses contained in the IP blocks listed in Google's ASN of 15169. For more information, see Implementing an HTTP Connection Server. Ensure that your firewall accepts outgoing connections from MobileFirst Server to android.googleapis.com on port 443.

**When testing with an emulator, use a Google Device

iOS

[screenshot] In the push services > push settings section of your application in the MFP operations console, add your APNS certificate and corresponding password.

> To learn about setting up push notifications required certificates, see this blog post: Understanding and setting up certificates required to use iOS devices and Apple Push Notifications services (APNS) (https://www.ibm.com/developerworks/community/blogs/worklight/entry/understanding-and-setting-up-push-notifications-in-development-evnironment?lang=en)

Apple Push Notifications Services

For push notifications to be sent, the following servers must be accessible from a MobileFirst Server instance: Sandbox servers: gateway.sandbox.push.apple.com:2195 feedback.sandbox.push.apple.com:2196

Production servers: gateway.push.apple.com:2195 Feedback.push.apple.com:2196

1-courier.push.apple.com 5223

APNS Certificate

- During the development phase, use the apns-certificate-sandbox.p12 sandbox certificate file.
- During the production phase, use the apns-certificate-production.p12 production certificate file.
- Place the certificate file either in the application root folder or in the application environment (iPhone or iPad) folder. The environment root folder takes the highest priority. *Note*: APNS certificates must have a non-blank password. *Note*: The APNS production certificate can only be tested once the application that utilizes it has been successfully submitted to the Apple App Store.

=== this overview tutorial should explain: - how to setup authenticated push; - prerequisite should be the security tutorial to understand the foundation of the new security model - maybe this should not be in this overview tutorial... maybe another set of tutorials limited to security in push?

=== The handling in client side tutorials should explain: - how to setup push notifications support in iOS Xcode project (editing the podfile?) - how to setup push notifications support in Andrid Studio project (editing the builde.gradle file?) - how to setup push notifications support in Cordova applications - how to intercept and display notifications in the client

=== The sending push notifications tutorials should explain: - the Push REST API - the send push console tab - and any server-side push API that can be used in adapters