

Handling Push Notifications in Cordova

fork and edit tutorial (<https://github.ibm.com/MFPSamples/DevCenter/tree/master/tutorials/en/foundation/8.0/notifications/handling-push-notifications-in-cordova.md>) | report issue (<https://github.ibm.com/MFPSamples/DevCenter/issues/new>)

Overview

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 by unsubscribing from the reserved `Push.all` tag.

Agenda

- Notifications Configuration
- Notifications API
- Handling a push notification
- Handling a secure push notification

Notifications Configuration

To get the application running for Android

1. Create cordova app using cordova create and mfp cordova template
2. `$ cordova platform add android`
3. `$ cordova platform add plugin cordova-plugin-mfp-push`
4. `$ cordova build`
5. Import the app/platforms/android into Android Studio
6. In `build.gradle(module:Android)`, add to repositories (2x)

```
maven {  
    url  
    "http://visustar.francelab.fr.ibm.com:8081/nexus/content/repositories/mobile-s"  
}
```
7. In `build.gradle(module:Android)`, add classpath 'com.google.gms:google-services:2.0.0-alpha3' to dependencies (3x)
8. In `build.gradle(module:Android)`, add `jcenter()` to repositories in buildscript block
9. Add compile 'com.google.android.gms:play-services-gcm:8.4.0' to `app/platforms/android/cordova-plugin-mfp-push/-build-extras.gradle` in dependencies
10. Add compile 'com.squareup.okhttp:okhttp:2.6.0' to `app/platforms/android/cordova-plugin-mfp-push/-build-extras.gradle` in dependencies
11. Add apply plugin: 'com.google.gms.google-services' to `app/platforms/android/cordova-plugin-mfp-`

push/-build-extras.gradle outside dependencies

12. Add google-services.json configuration file to app/platforms/android folder
13. Change version to '8.0.0-Beta1-SNAPSHOT' in app/platforms/android folder
14. Add the Push SDK APIs to your application (Refer the sample application)
15. If you want to change the notification title, then add push *notification*tile in strings.xml

To get the application running for iOS

1. Create Cordova project without using cordova mfp template
2. \$ cordova platform add ios
3. \$ cordova platform add plugin cordova-plugin-mfp-push
4. \$ cordova build
5. Open in XCode
6. Use the Push SDK APIs (Refer Sample)

Notifications API

API methods for tag notifications

Client-side API

- `MFPPush.subscribeTag(tagName,options)` - Subscribes the device to the specified tag name.
- `MFPPush.unsubscribeTag(tagName,options)` - Unsubscribes the device from the specified tag name.
- `MFPPush.registerDevice(options)` - Registers devices for push notifications (?!? confirm definition ?!?)
- `MFPPush.isPushSupported()` - Returns `true` if push notifications are supported by the platform, or `false` otherwise.
- `MFPPush.isTagSubscribed(tagName)` - Returns whether the device is subscribed to a specified tag name.

Common API methods for tag and broadcast notifications

Client-side API

- `WL.Client.Push.onMessage (props, payload)` - This method is called when a push notification is received by the device.
- **props** - A JSON block that contains the notification 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`.

```

WL.Client.Push.onMessage = function (props, payload) {
  WL.Client.Push.onMessage = function (props, payload) {
    WL.SimpleDialog.show("Tag Notifications", "Provider notification data: " + JSON.stringify(props), [ {
      text : 'Close',
      handler : function() {
        WL.SimpleDialog.show("Tag Notifications", "Application notification data: " + JSON.stringify(payload), [ {
          text : 'Close',
          handler : function() {}
        }]);
      }
    }]);
  }
};
}

```

Handling a push notification

Handling a secure push notification