Push Notifications

Agenda

• What is Push Notification
• Short history
• How it works
• Types of Push Notifications & Vendors
• How to implement a push notification using Urban Airship & Firebase
Push notifications let your application notify a user of new messages or events even when the user is not actively using your application.
Short history

- C2DM (Cloud to device messaging) launched in 2010
- C2DM is officially deprecated on June 26th 2012 and shutdown on Oct 20, 2015
- “GCM” Google cloud messaging replaced C2DM
  - Battery efficiency & rich set of new APIs.
- GCM’s latest version is “FCM” Firebase cloud messaging.
  - Simple client development & out of box notification solution.
Push Notifications

How it works

source: https://developers.google.com/cloud-messaging/gcm
How it works (in detail)
Types of Push Notifications

- Location based notifications
- Time based notifications
- General notifications
- In-app notification
- In-app message
Vendors

- Firebase
- Push IO
- SessionM
- Urban Airship
- Vibes (uses Xtify internally)
- Xtify (IBM product)
Implementation steps (For UA)

- Setup Google project
- Setup Urban Airship SDK
- Configure project Manifest file
- Configure Urban airship properties
- Enable Push notifications for the app.
Setup Google project

- Create a project in Google Developers Console at https://console.developers.google.com

- Turn GCM for android ON.

- Generate an API Key
  - Create new Key in API & auth
  - Select server key and click create without entering any IP
Setup Google project

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

Add credentials
Setup Google project

You need credentials to access APIs. Enable the APIs you plan to use and then create the credentials they require. Depending on the API, you need an API key, a service account, or an OAuth 2.0 client ID. Refer to the API documentation for details.

- **API key**
  Identifies your project using a simple API key to check quota and access. For APIs like Google Translate.

- **OAuth 2.0 client ID**
  Requests user consent so your app can access the user's data. For APIs like Google Calendar.

- **Service account**
  Enables server-to-server, app-level authentication using robot accounts. For use with Google Cloud APIs.
Setup Urban Airship SDK

- Open your android studio project
- Add compile project (`urbanairship-lib-x.x.x`) to dependencies in build.gradle
- Cardview library, google play service GCM and Location (ver 7.0 or newer) are other dependencies.
- Don’t forget to sync the project after updating build.gradle
Setup Urban Airship SDK

Sample build.gradle dependencies

dependencies {
    // Urban Airship SDK
    compile 'com.urbanairship.android:urbanairship-sdk:6.1.+'

    // Recommended for in-app messaging
    compile 'com.android.support:cardview-v7:22.2.0'

    // Recommended for location services
    compile 'com.google.android.gms:play-services-location:7.5.0'

    // Required for Android (GCM) push notifications
    compile 'com.google.android.gms:play-services-gcm:7.5.0'
}

Modify Manifest file

Sample manifest.xml

```xml
<!-- REQUIRED for Urban Airship -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<!-- Keeps the processor from sleeping when a message is received. -->
<uses-permission android:name="android.permission.WAKE_LOCK" />
<uses-permission android:name="android.permission.VIBRATE" />
<!-- Required for Push -->

<permission android:name="${applicationId}.permission.UA_DATA" android:protectionLevel="signature" />
<uses-permission android:name="${applicationId}.permission.UA_DATA" />
<!-- The two elements above ensure that only this application has access to the Urban Airship provider -->

<!-- OPTIONAL Urban Airship Settings -->
<!-- REQUIRED FOR LOCATION -->
<!-- Use ACCESS_COARSE_LOCATION if GPS access is not necessary -->
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```
Modify Manifest file (continue)

Sample manifest.xml

```xml
<!-- REQUIRED for Action.startActivityForResult -->
<activity android:name="com.urbanairship.actions.ActionActivity" />

<!-- REQUIRED for Urban Airship Push -->
<activity android:name="com.urbanairship.CoreActivity" />

<!-- REQUIRED for Urban Airship Push. The priority is important to be set lower than the application’s push intent receiver in order for the push intent receiver to handle push intents before the core receiver. This allows the application to launch any activities before Urban Airship performs any actions or falls back to launching the application launch intent. -->
<receiver android:name="com.urbanairship.CoreReceiver"
    android:exported="false">
    <intent-filter android:pRIORITY="-999">
        <action android:name="com.urbanairship.push.OPENED" />
        <category android:name="${applicationId}" />
    </intent-filter>
</receiver>
```
Modify Manifest file (continue)

Sample manifest.xml

```xml
<!-- REQUIRED for Landing Pages
    - For more customization details, see com.urbanairship.actions.LandingPageActivity -->
<activity
    android:name="com.urbanairship.actions.LandingPageActivity"
    android:exported="false">
    <intent-filter>
        <action android:name="com.urbanairship.actions.SHOW_LANDING_PAGE_INTENT_ACTION" />
        <data android:scheme="http" />
        <data android:scheme="https" />
        <data android:scheme="message" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
</activity>

<!-- REQUIRED for Urban Airship -->

<!-- Required for analytics -->
<service android:name="com.urbanairship.analytics.EventService" android:label="Event Service" />
```
Create a new `airshipconfig.properties` file and place it in your project’s assets folder.

```properties
developmentAppKey = Your Development App Key
developmentAppSecret = Your Development App Secret

productionAppKey = Your Production App Key
productionAppSecret = Your Production Secret

# Toggles between the development and production app credentials
# Before submitting your application to an app store set to true
inProduction = false

# LogLevel is "VERBOSE", "DEBUG", "INFO", "WARN", "ERROR" or "ASSERT"
developmentLogLevel = DEBUG
productionLogLevel = ERROR
```
In your Application object’s `onCreate()`

```java
@Override
public void onCreate() {
    UAirship.takeOff(this);
}
```

Above code will bootstrap the SDK and look for the settings in `airshipconfig.properties`.

This can also optionally be provided at run time by passing an `AirshipConfigOptions` in the call to `com.urbanairship.UAirship#takeOff(Application, AirshipConfigOptions)`:

```java
@Override
public void onCreate() {
    AirshipConfigOptions options = new AirshipConfigOptions();
    options.developmentAppKey = "Your Development App Key";
    options.developmentAppSecret = "Your Development App Secret";
    options.productionAppKey = "Your Production App Key";
    options.productionAppSecret = "Your Production App Secret";
    options.inProduction = false;

    UAirship.takeOff(this, options);
```
Enabling user notification

After user’s consent enable notification for your app

```java
UAirship.shared().getPushManager().setUserNotificationsEnabled(true);
```

The Channel ID is a unique identifier that ties together an application/device pair on Android devices. The Channel ID is used to target pushes to specific devices using the Urban Airship API. Once a Channel ID is created, it will persist the application until it is either reinstalled, or its internal data is cleared.

```java
String channelId = UAirship.shared().getPushManager().getChannelId();
Logger.info("My Application Channel ID: " + channelId);
```
App is all ready for receiving push notifications
Push Notifications

Urban Airship Dashboard

Notification Services
Set up third-party services for integration with our platform.

- **Apple Push Notification Service (APNS)**
  - Apple's service for notifications on iOS devices.

- **Google Cloud Messaging (GCM)**
  - Google's service for notifications on Android devices.
  - **Api key**: AtzaSyDutLkJcOkO13Dk8F7e14LjbrGj6U
  - **Package**: com.bottlerocket.sai.myapplication

- **Amazon Device Messaging**
  - Amazon's notifications service.

- **Microsoft Windows Push Notification Service (WNS)**
  - Microsoft's service for toast, tile, and badge updates on Windows 8 devices.

- **Microsoft Push Notification System (MPNS)**
  - Microsoft's service for toast, tile, and badge updates on Windows Phone 8 devices.

- **Blackberry Push Notifications**
  - Blackberry Push Notifications for blackberry devices.
Push Notifications

Urban Airship Dashboard

New Notification

Message

This is a test Notification

You have 209 bytes left.

Platforms and Recipients

Android

- Broadcast
- Devices by Tag
- One Device
- Devices by Segment

This means all devices that have registered with this application.

Delivery Time

- Right Now
- Later

Optional Settings

Send Notifications
Urban Airship dashboard
Firebase architecture

source: https://firebase.google.com/docs/cloud-messaging/
Implementation steps for Firebase

- Setup Firebase and FCM SDK

- Optional for extra feature.
  - A service that extends FirebaseMessagingService
  - A service that extends FirebaseInstanceIdService

minSdkVersion 8
Implementation steps for Firebase

Code sample & Demo
Questions?
References

- http://docs.urbanairship.com/platform/android.html
- https://developers.google.com/android/guides/setup
- https://developers.google.com/cloud-messaging/gcm
- http://docs.urbanairship.com/user-guide/getting-started.html#ug-sending-your-first-push
- https://firebase.google.com
- https://developers.google.com/cloud-messaging/faq
Thank you!