Setting up Urban Airship

14/10/2013

Push is one of the things you should add to any app, even if at first you'd only use it for broadcast messages. It's the best way to keep in touch with your users, even when – or exactly because – they haven't used your app for months.

Urban Airport

To make this a no-brainer I've written an CommonJS module that wraps the iOS and Android versions of Appcelerator's open-source UrbanAirship module (https://marketplace.appcelerator.com/apps/4984), and is compatible with both Alloy and Classic projects. It hides you from all implementation differences in registering for and receiving push notifications.

The module is available via my UTiL repo (https://github.com/FokkeZB/UTiL/tree/master/urbanairport).

Walk-through

Let me take you through the exact steps of setting up push via Urban Airship with this wrapper:

1. Sign up for the free developer edition (https://go.urbanairship.com/accounts/register/) of Urban Airship.
2. Create two apps of which one you add a *DEV* suffix to the *Application Name* and select *Development* as *Production Status.*
3. Follow the Set Up Your Application With Apple (http://docs.urbanairship.com/build/ios.html#set-up-your-application-with-apple) paragraph of the UA iOS Getting Started guide.
4. Follow the Google Setup (http://docs.urbanairship.com/build/android.html#google-setup) paragraph of the UA Android Getting Started guide.
5. Download the latest Android (https://github.com/appcelerator/titanium_modules/tree/master and iOS (https://github.com/appcelerator/titanium_modules/tree/master for `.zip`) version of Appcelerator's module and extract them to your project's modules folder or the global `~/Library/Application Support/Titanium/modules` folder.
6. For both the Android (https://github.com/appcelerator/titanium_modules/tree/master and iOS (https://github.com/appcelerator/titanium_modules/tree/master module copy the file in the module's `example/platform` folder to the exact same location under your project.
7. Set the development and production keys and secrets in both files. Get them from the Urban Airship dashboard (https://go.urbanairship.com/apps/) under *Settings > API Keys* for both of the 2 apps you have created.
8. For Android, in the same `airshipconfig.properties` file, set transport to `gcm` and `gcmSender` to the Project Number that can be found (https://code.google.com/apis/console/) on the *Overview* tab of the project you created at step 4.

9. Download the urbanairport (https://github.com/FokkeZB/UTiL/tree/master/urbanairport) wrapper and place it in your project's `app/lib` or `Resources` if you have use Titanium Classic.
10. In your `alloy.js` or classic `app.js` initialize the wrapper:

```
var urbanairport = require('urbanairport');

urbanairport.register({
  debug: true, // Show debug info

  // Sets push types
  sound: true,   // iOS + Android (default)
  vibrate: true, // Android (default)
  badge: true,   // iOS (default)
  alert: true,   // iOS (default)

  // Set any property and call any single-property m
  autoBadge: false,

  // Enable compatibility mode (see below)
  compatibility: true,

  // On Android these will be automatically set once
  alias: 'John',
  tags: 'single', // Supports both a single or Array

  callback: function(e) { // The only callback you r

    // Registration failed
    if (e.type === 'error') {
      alert('Sorry, no push for you: ' + e.error);

    // Registration done
    } else if (e.type === 'success') {
      alert('Your token is: ' + e.deviceToken);
```

```
31
32      // Received notification
33      } else if (e.type === 'callback') {
34
35        // Properties are normalized for iOS and Andro
36        // e.payload === e.data === e.data.aps
37        // e.message === e.data.alert === e.data.aps.a
38        alert(e.message);
39      }
40    }
41 });
42
43 // Manually disable/re-enable push
44 urbanairport.disable(); // enable();
45
46 // Append tags instead of resetting them
47 urbanairport.addTags('foo'); // Both single and Arra
48
49 // Set any property and call any single-property met
50 urbanairport.showOnAppClick = true;
```

Compatibility mode

It's important to be aware of **when** the callback will be called with `e.type === 'callback'`. On both iOS and Android this will only get called when the app is running. Only on iOS the app needs to run in the foreground or be called to the foreground by tapping on the notification. On Android the callback will be fired twice for the same notification if the app was running (in background) when it was received and then the app was opened because the user tapped on it. The second time `e.clicked` will be `TRUE`, but since the app might also **not** have been running when the notification was received, you cannot just ignore calls with this value.

To make up for this, you can use `compatibility: TRUE` with the `register` method. This will trigger the wrapper to do 2 things:

1. Default `showOnAppClick` to `TRUE` so that tapping the notification will open the app, just like on iOS.
2. Create a hash for each notification and only call the callback the first time. Pretty much like on iOS, but with the advantage of it being called also when the app is in the background.

Share