Build The Android App

This document describes how to build the Ushahidi Android app from source. Instructions are provided below for building with both Eclipse and on the command line with Ant.

You will require the following to compile and run the app (instructions for these are not included in this document):

- Java 1.6
- git
- Android SDK (API 7 or later)

Getting the source code

Clone the source from github. There are different branches in the git repository. The Branch structure section in the README file explains the different branches that are there.

On the command line, issue the command below to clone the repository.

```
git clone git://github.com/ushahidi/Ushahidi_Android.git
```

Setting up a Google Maps API Key

Since the Ushahidi Android app is heavily based on maps, you'll have to obtain an API Key for the mapping functionality to work properly. Google Maps Android v2 requires setting up the Google Play services SDK. This has already be done for you. You don't need to repeat the steps to set that up. The process of obtaining and adding an API key is as follows:

- Obtain an API Key.
- Add the obtained API key to Themes/Ushahidi/res/values/theme.xml file. Replace the value of google_map_api_key with the new API key. Eg.

```
<string name="google_map_api_key">the_new_google_maps_api_key</string>
```

Building with Eclipse

1. Setup Core Library project

The Core project depends on three main libraries. These libraries come with the Ushahidi Android source. They are located in the Libraries/ folder

- abs - Actionbar Sherlock library provides action bar functionality for pre-honeycomb devices.
- menudrawer - Provides the slide out / slide in menus
- google-play-services_lib - Provides Google Maps Android v2 support.

In Eclipse,

- Import Core project into your workspace.
- Import abs library into your workspace.
- Import google-play-services_lib into your workspace
- Add ushahidi_sdk-xx.jar to your project build path. This is located in Core/libs.
    - To add jar file to your build path, Right-click the JAR file in the Package Explorer, select Build Path > Add to Build Path

Note: If you fail to import any of the libraries, the build process will fail miserably.

2. Setup Ushahidi project.

You can repeat this step for the other white-labelled apps.

- Import Ushahidi from Themes/Ushahidi into your workspace.
- Reference the imported Core library in the imported Ushahidi project. Have a look at http://developer.android.com/tools/projects/projects-eclipse.html#ReferencingLibraryProject for a comprehensive guide on how-to reference a project library.

Now build and run Ushahidi project not the Core library project.

Building on the Command Line

1. Setup Library Projects

The Core project depends on the libraries found in Libraries/. These include:

- abs - Actionbar Sherlock library provides action bar functionality for pre-honeycomb devices.
- google-play-services_lib - Provides Google Maps Android v2 support.

For each of these sub-directories run the Android update library project command. For example:

```
$> android update lib-project -p <directory_name> -t <num>
```

Where <directory_name> is the library's directory, and <num> is the target number for the API level you're building for.

More details on referencing library projects when building on the command line can be found at: http://developer.android.com/tools/projects/projects-cmdline.html#ReferencingLibraryProject

2. Setup the Core Library Project

The Core project is also a library project. Run the same command as shown above on the Core/ directory.

3. Setup the Ushahidi Project

The app is built from one of the sub-directories in Themes/. Assuming you are going to use the default theme found in Themes/Ushahidi/ do the following:

- Create an empty sub-directory src/ under Themes/Ushahidi. The build.xml file from the Android SDK expects this directory to exist for all Android projects.
- Run the Android update project command as follows:

```
$> android update project -p <path to themes/ushahidi> -t <num>
```

Now build using the ant command inside the Ushahidi project, not the Core project.

Note: You'll need an Android device to run the application as Google doesn't support "Google Maps v2 Android" on the emulators.