

1. Setup project

Assuming that you already have installed:

Ionic 2.1.8

Cordova

Typescript

Android SDK (to run android app)

Xcode (to run ios apps)

Create blank ionic 2.1.8 project in typescript.

```
ionic start StarterFirebasePackV2 blank --v2
```

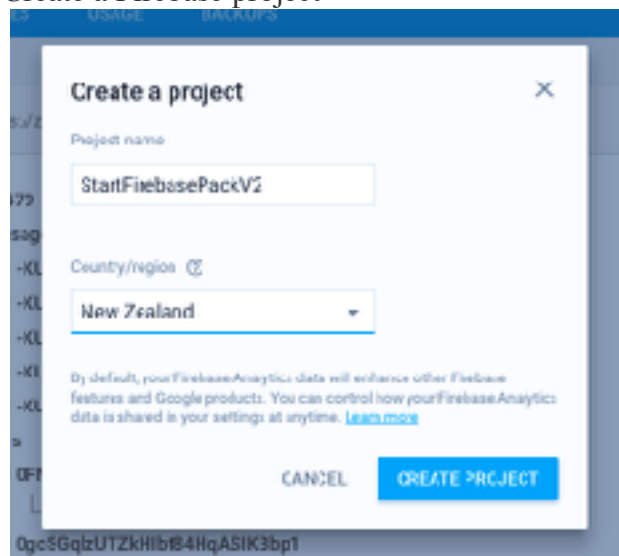
Copy the folders `src`, `package.json` and replace them in your new project root.

Run `npm install` inside the project folder. To install all dependencies added to the new `package.json` file.

Run `ionic state restore` inside the project folder. To install all plugins added to the new `package.json` file.

2. Configuring Firebase project

Create a Firebase project



Add project to Web application.



Add info inside the `config.ts`

```
export const firebaseConfig = {
```

```

apiKey: "",
authDomain: "",
databaseURL: "",
storageBucket: ""
};

```

3. Anonymous and Email/password auth.

Enable Anonymous and Email/Password authentications in Firebase.



If you run `ionic serve` in your project root, you should be able to Register and authenticate via Email/password and Anonymous users.

4. Facebook OAuth v4 SDK

Enable Facebook login in Firebase – **Copy the redirect URL since we will use it later, leave this page opened for now, we will get the App ID and App secret later.** Create Facebook developer app, use the Basic Setup.

<https://developers.facebook.com/apps>

Save, grab the App ID and App Secret in your Facebook Dashboard, and add them in the Firebase Facebook Authentication.

Run the following command:

```
ionic plugin add cordova-plugin-facebook4 --save --variable APP_ID="facebook_app_id" --variable APP_NAME="StarterFirebasePackV2"
```

At this point the Facebook login should work in the Browser Mode only. (See How to activate/deactivate Browser Mode in the last step of this document)

However, in order to make the Facebook login work for iOS and Android devices, we need an additional step.

Let's enable for iOS first.

In the Facebook developer dashboard, go to Settings.

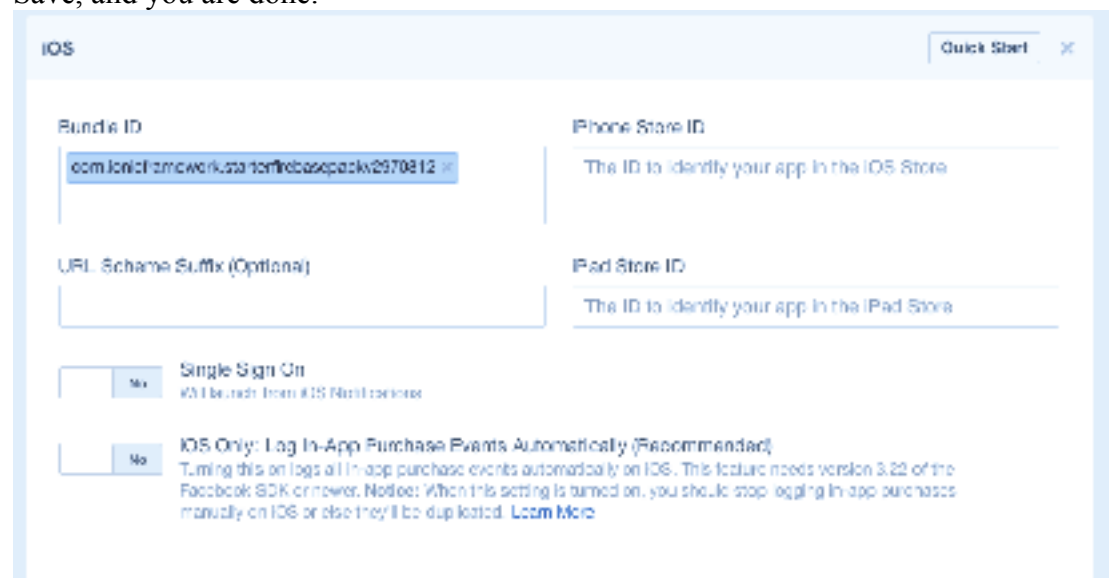
Add a new platform.

Choose iOS.

Now add your Bundle ID – found in the config.xml file.

<widget id=.....>

Save, and you are done.

A screenshot of the Facebook iOS Settings form. The form is titled "iOS" and has a "Quick Start" button with a close icon in the top right corner. It contains several input fields and checkboxes. The "Bundle ID" field is filled with "com.ionicframework.starterfirebasepackv2970812" and has a small 'x' icon to its right. Below it is the "URL Scheme Suffix (Optional)" field. To the right of the Bundle ID field is the "Phone Store ID" field with a description: "The ID to identify your app in the iOS Store". Below the URL Scheme Suffix field is the "iPad Store ID" field with a description: "The ID to identify your app in the iPad Store". At the bottom, there are two checkboxes. The first is "Single Sign On" with a "No" button and a note: "Will bundle from all iOS Platform versions". The second is "iOS Only: Log In-App Purchase Events Automatically (Recommended)" with a "No" button and a note: "Turning this on logs all in-app purchase events automatically on iOS. This feature needs version 3.22 of the Facebook SDK or newer. Note: When this setting is turned on, you should stop logging in-app purchases manually on iOS or else they'll be duplicated. Learn More".

Next up Android.

In the Facebook developer dashboard, go to Settings.

Add a new platform.

Choose Android.

Add the same value for the Bundle ID in iOS in the field Google Play Package Name.

Now things gets a little bit nasty, we need to generate a Hash key for the Android apk.

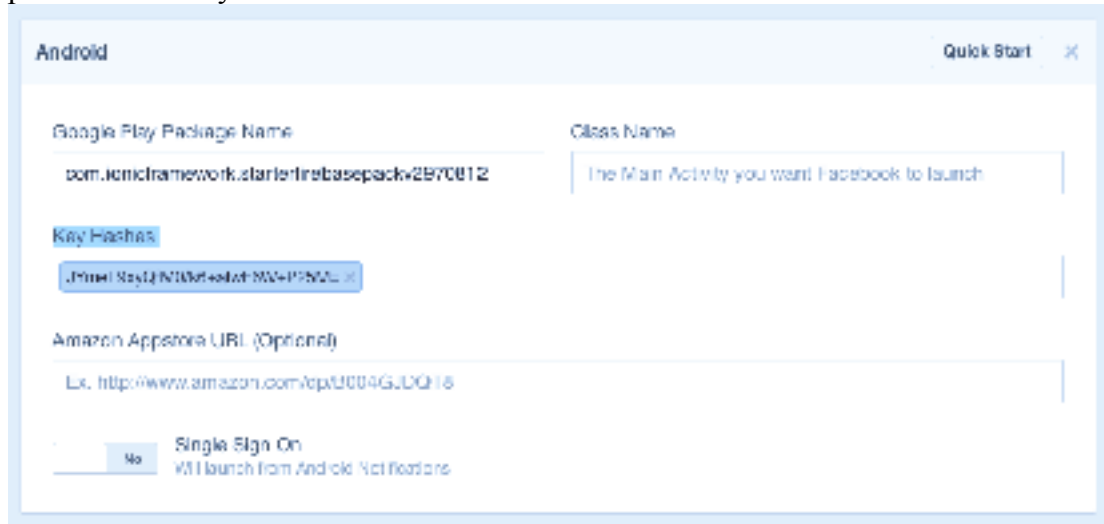
We are going to generate a Hash for the debug apk, so we can deploy our app to an Android device and test.

First thing, run the command in your project root folder.

```
keytool -exportcert -alias androiddebugkey -keystore ~/.android/debug.keystore |  
openssl sha1 -binary | openssl base64
```

Password for the debug.keystore should be always *android*.

Now copy the hash generated by this command and paste into the Facebook Android product field Key hashes.

A screenshot of the Facebook Android app configuration interface. The window has a title bar with 'Android' on the left and 'Quick Start' with a close button on the right. The main content area has a light blue background. It contains several input fields: 'Google Play Package Name' with the value 'com.ionicframework.startertirebasepack2870012', 'Class Name' with the value 'The Main Activity you want Facebook to launch', 'Key Hashes' with a blue button containing the hash 'JPMHl8x1QjN10Wt+stWf9W4P2NMU...', and 'Amazon Appstore URL (Optional)' with the value 'Ex. http://www.amazon.com/tp/U004GJLQJ18'. At the bottom, there is a 'Single Sign On' section with a 'No' button and the text 'Will launch from Android Notifications'.

And we are done!

You should be able to Login via Facebook now using devices. (Disable the Browser Mode to test the app in a real device - See How to activate/deactivate Browser Mode in the last step of this document.)

5. Twitter oAuth

Enable Twitter authentication if Firebase.

Copy the callback URL.

Create a Twitter application at: <https://apps.twitter.com/app>

Add the callback URL to the application.

Create an application

Application Details

Name *
GatedFirebaseAuth2
Your application name. This is used in authenticating the user and in displaying authentication errors. 30 characters max.

Description *
Your application description, which will be shown in user-facing authentication screens. Between 10 and 200 characters max.

Website *
www.gatedfirebase.com
Your application's publicly accessible base URL. When users can't connect, make use of or find out more information about your application. This fully-qualified URL is used in the source attribution for events received by your application and will be shown in user-facing authentication screens. (If you're not using a base URL, just put your domain name without a trailing slash.)

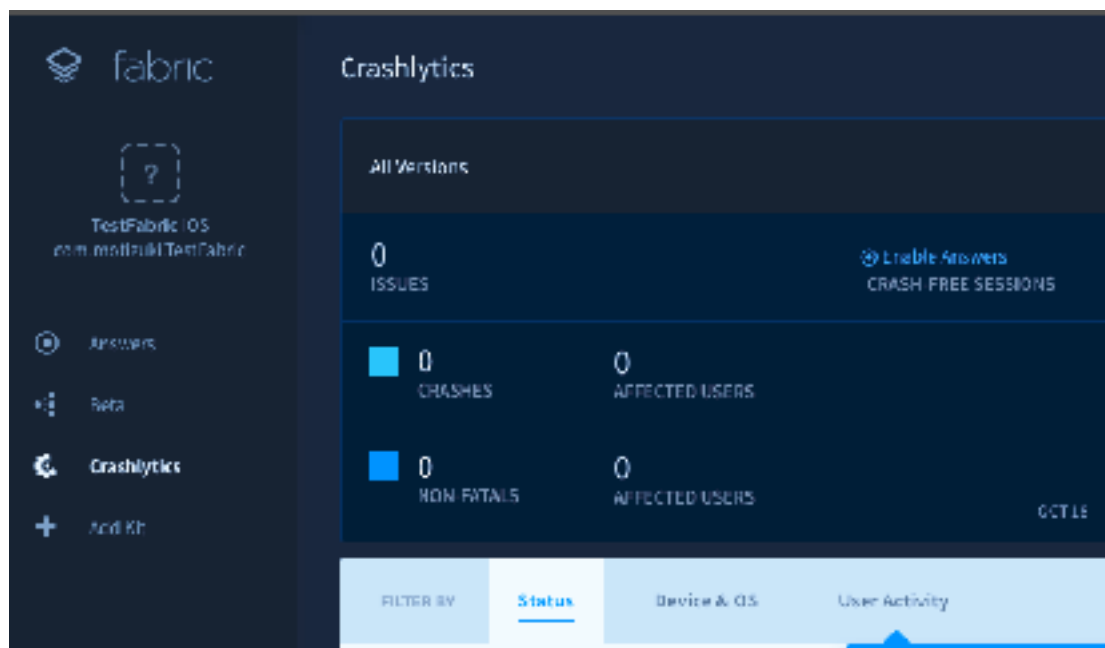
Callback URL
https://site.firebaseio.com/v2/firebase/auth/callback?authRedirect=true
Where should we return after successfully authenticating? OAuth 1.0 applications should explicitly specify their callback URL, or the browser token dialog, regardless of the value given here. To access your application's data using OAuth2, you must use this URL.

Go to keys and access token tab and grab your API key and API secret.
Go back to Firebase website and add them to the Twitter Authentication and save.

Now we need to create a Fabric application to use the Fabric API in the Twitter Connect Plugin.

Go to : <https://fabric.io>

Create any project, It could be an iOS application for instance.



The only thing we really need is the API key. Getting the API key is fairly tricky, but this process seems to work:

- Login to Fabric account and open <https://fabric.io/kits/android/crashlytics/install>

- Find the meta-data code block in AndroidManifest.xml
- Find your API Key pre filled in the code.

Now run the following command with your **Fabric API KEY** in your project root:
`cordova plugin add twitter-connect-plugin --variable FABRIC_KEY=<Fabric API Key>`

The last thing you need to do now is to open config.xml (in your project's root) and add these two lines before the closing </widget> tag:

```
<preference name="TwitterConsumerKey" value="<Twitter Consumer Key>" />
<preference name="TwitterConsumerSecret" value="<Twitter Consumer Secret>" />
```

If you get stuck in one of those steps, you can checkout the plugin github page for more information: <https://github.com/ManifestWebDesign/twitter-connect-plugin>.
 Done.

6. GooglePlus oAuth

Enable Google authentication in Firebase.

Setup iOS:

To get your iOS REVERSED_CLIENT_ID, generate a configuration file here. This GoogleService-Info.plist file contains the REVERSED_CLIENT_ID you'll need during installation.

Go to: <https://developers.google.com/mobile/add?platform=ios&cntapi=signin>

Create a new application by passing a name and adding the BundleID.

Enable Google Sign in.

Download the plist file.

You will find your reverse client ID inside this file.

Run the following command using your reverse client id:

```
cordova plugin add cordova-plugin-googleplus --save --variable REVERSED_CLIENT_ID=myreversedclientid
```

Setup Android:

Go to: <https://developers.google.com/mobile/add?platform=android&cntapi=signin>

Create a new application by passing a name and adding the BundleID.

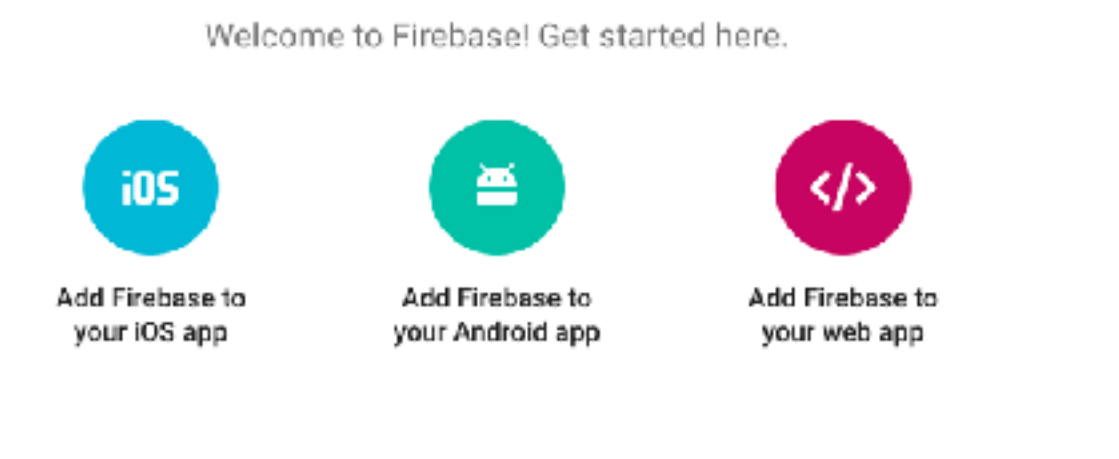
Enable Google Sign in.

We need to get the SHA1 Hash now.

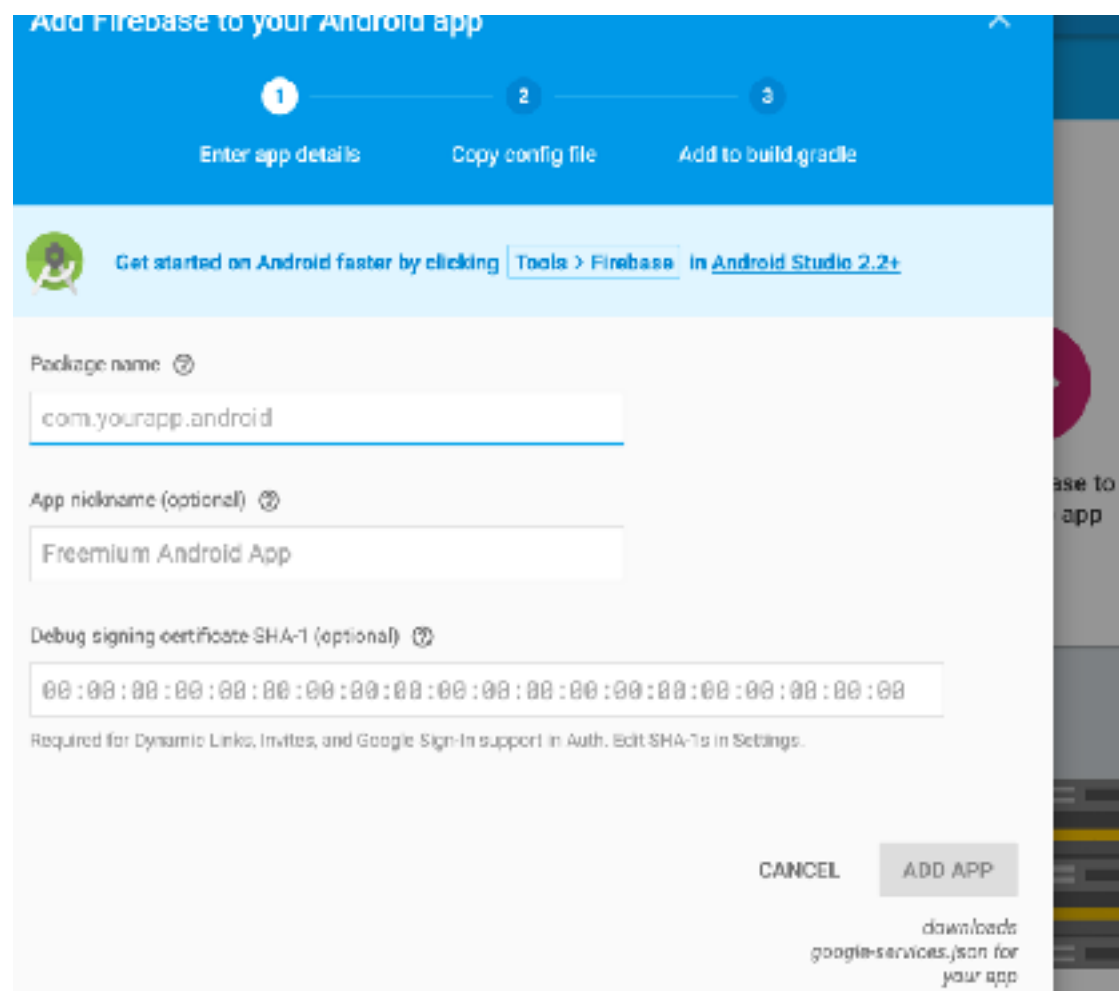
Run the command:

```
keytool -exportcert -list -v \
-alias androiddebugkey -keystore ~/.android/debug.keystore
```

Get the SHA1 and add to the enable the Google Sign in.



Add firebase android app.



Add the SHA-1.

Lastly, get your web client ID at:

<https://console.developers.google.com/apis/credentials>

Add it to the /src/app/config.ts

Make sure all client ids were correctly added, and they should all be present in the console dashboard.

Dashboard	Name	Creation date	Revision	Key
Library	Android key (auto created by Google Service)	Oct 19, 2016	None	AizaSy2Ac5FF1
Credentials	iOS key (auto created by Google Service)	Oct 19, 2016	None	AizaSy2BFm9W

OAuth 2.0 client IDs				
Name	Creation date	Type	Client ID	
Android client for com.ionicframework.ionic26257384 (auto created by Google Service)	Oct 19, 2016	Android	3721841-ws2g941	
iOS client for com.ionicframework.ionic26257384 (auto created by Google Service)	Oct 19, 2016	iOS	3721841	
Web client (auto created by Google Service)	Oct 19, 2016	Web application	3721841	

If you had any trouble running those steps, checkout the github website for the plugin:

<https://github.com/EddyVerbruggen/cordova-plugin-googleplus>

7. Enable / Disable Browser mode

With the new changes in the oAuth plugins if you want to play around with your application in the browser we need to change how the oAuth method is implemented.

Go to the file /src/app/authentication/pages/login/login.ts in the line 61.

These are the variations:

User this one to run the app in a real device

`this.authenticator.signInWithOAuth(provider)`

This one to run in the browser (ionic serve)

`this.authenticator.signInWithOAuthBrowserMode(provider)`

By doing this the oAuth will work in both environments as you need.

Feel free to contact me if you have any questions.

gustavokm90@gmail.com

Happy coding