May 11 2017 - Updated to Ionic v3

We know an app without functionalities is like a car without wheels so we have created this version of Ion2FullApp with the following integrations.
You can find the technical documentation for each of these integrations in the following pages.
Enjoy :-)

Important notes before you start:

1. Please before start read the SET UP section
2. For the general Ion2FullApp documentation please go here: bit.ly/ionicthemes-ion2fullapp
3. Please note that some cordova plugins only work in the device or emulator, not in the browser. So please try to test the app on the device or emulator.

Facebook Connect

We have integrated the possibility to login and signup with Facebook app.
You can find all the code needed for this in: `src/pages/facebook-login`

In order to provide your users with the possibility to connect your app with Facebook, you need to create a Facebook App and get an APP_ID and APP_NAME. Follow the STEP 1 from this link.

After you have created you Facebook app it’s time to install the plugin into your app. Follow these steps to get this DONE:

- Execute the following command on the terminal changing the variables with your own values. Replace APP_ID and APP_NAME variables with your Facebook app values

```
$ ionic plugin add cordova-plugin-facebook4 --variable APP_ID="123456789" --variable
APP_NAME="myApplication" --save

npm install --save @ionic-native/facebook
```

Then in **src/pages/facebook-login/facebook-login.service.ts** set your **APP_ID** to the variable: **FB_APP_ID**

```
FB_APP_ID: number = 123456789;
```

If you want to get more information from the user you need to have the correct permissions. You can read here which permissions do you need for each action.

In `src/pages/facebook-login/facebook-login.service.ts` inside the method `doFacebookLogin()` you will find the following line where you have to include the permissions you need.

```
Facebook.login(["public_profile"]).then(...
```

`["public_profile"]` is the array of permissions, you can add more if you need.

We save the user data in the NativeStorage, but you may want to save it in your database.

When you go to Functionalities/Facebook Integration in the app you will see a screen with the data extracted from Facebook. Please note that in this example the friends and photos lists are populated with dummy data.

Google Connect

We have integrated the possibility to login and signup with Google+.
You can find all the code needed for this in: `src/pages/google-login`

In order to provide your users with the possibility of log into your app with Google+, you need some configurations. Follow the STEP 1 form this link

After you have done the step 1 from above, you just need to install the plugin into your app.
Follow these steps to get this done:

- Execute the following command on the terminal changing the variable `REVERSED_CLIENT_ID` with your own values.

```
$ cordova plugin add cordova-plugin-googleplus --variable
REVERSED_CLIENT_ID=com.googleusercontent.apps.your_reversed_clientid

$ npm install --save @ionic-native/google-plus
```

Then in **src/pages/google-login/google-login.service.ts** set your **webClientId** to the variable: **webClientId**

You can get your **webClientId** in your project Developer's Console.

```
webClientId: string = "yourID.apps.googleusercontent.com";
```

If you want to get more information from the user you need to have the correct scopes.
The default scopes requested are profile and email. To request other scopes, add them as a space-separated list to the scopes parameter. They will be requested exactly as passed in. Refer to the Google Scopes documentation for info on valid scopes that can be requested. For example, 'scope': 'https://www.googleapis.com/auth/youtube or https://www.googleapis.com/auth/tasks'.

Naturally, in order to use any additional scopes or APIs, they will need to be activated in your project Developer's Console.

We save the user data in the NativeStorage, but you may want to save it in your database.

When you go to Functionalities/Google Integration in the app you will see a screen with the data extracted from Google+. Please note that in this example the friends and photos lists are populated with dummy data.

Social Sharing

We use this plugin to give the app the power to share text, files, images, and links via social networks, sms, and email.

To install run:

```
$ ionic plugin add cordova-plugin-x-socialsharing --save

$ npm install --save @ionic-native/social-sharing
```

This plugin is super easy to use and to install.

https://github.com/EddyVerbruggen/SocialSharing-PhoneGap-Plugin#4a-usage-on-ios-and-android

feed.html

```
<button class="action-button" ion-button clear small (click)="sharePost(post)" icon-left>
     <ion-icon name='share-alt'></ion-icon>Share
</button>
```

feed.ts

```
sharePost(post) {
  //this code is to use the social sharing plugin
  // message, subject, file, url
  SocialSharing.share(post.description, post.title, post.image)
  .then(() => {
     console.log('Success!');
  })
  .catch(() => {
     console.log('Error');
  });
}
```

Google Maps

Installation

To use Google Maps **Javascript SDK** we have to add this line in `index.html`

```
<script src="https://maps.googleapis.com/maps/api/js?v=3.exp&libraries=places"></script>
```

Then we have to install the following typing:

```
npm install @types/googlemaps --save
```

Note: if you did the above configurations and you are still getting the message:
"Cannot find namespace Google" try doing the following:

1. Install the TypeScript Definition Manager CLI

```
npm install typings --global
```

2. Then use your CLI to navigate into your Ionic 2 project's folder
3. Install the google map definition files using the typings CLI

```
typings install dt~google.maps --global
```

There are two ways to implement Google Maps:

1. Web based - Javascript
2. Native

But what's the difference between these two options and which one is better?
There are a few points we should consider:

Javascript SDK

This option uses more mobile data, and the experience won't be as smooth as the native version. As the user views a map in this web based version, data tiles in .png format are loaded in on the fly. This is worse for data consumption and can cause the user to have to wait longer for chunks of the map to load in. To learn more about this option please check out this article.

Native SDKs

This option requires Cordova, so will only run when installed on devices, not through the browser. If you only need to support iOS and Android then this isn't a problem, but with the rising popularity of progressive web apps you may want to have your app available through the mobile web as well.

Conclusions

Finally, the Native SDK Cordova plugin is an abstraction on top of the individual iOS and Android SDKs, which allows us to interact with the SDKs from our mobile app, but also adds an extra layer where bugs can creep in. There are quite a few known issues with this plugin. When you use the Javascript SDK you are using the code directly from Google, so it is the more stable option.

Scored on those points above, it'd be a tie between the JavaScript SDK and the Native SDK, so the answer to the question of "which one is better?" would be the ever popular: it depends. You'll have to consider the requirements of your application and make a decision based on that.

In this app we decided to use the Javascript SDK because we have an open mind about progressive web apps feature.

In order to interact with this Google Maps library properly from angular we created the following elements:

Directive

We created a directive **`<google-map>`** to display the map and define a bridge to the google.maps.Map object. You can find it in `src/components/google-map/google-map.ts`

Service

We created a service to interact with all google maps API's we used in the example. You can find it in `src/pages/maps/maps.service.ts`

- **google.maps.places.AutocompleteService**
  - Provides places suggestions for the search and autocomplete feature
- **google.maps.Geocoder**
  - Handles the geocoding. This means getting the location (lat, lng) from a place and vice-versa.
- **google.maps.places.PlacesService**
  - To search and display nearby places
- **google.maps.DirectionsService**
  - To get and display directions from point A to point B
- **google.maps.DistanceMatrixService**
  - To get distances between two or more points

Models

In order to handle all these functionalities clearly in one page we created some angular models to helps us get things tidy and understandable. You can find them in `src/pages/maps/maps.model.ts`

NgZones

One caveat worth mentioning here is that there are known issues with external libraries interacting with the angular zone. In this particular case the issue is because the google.maps API's are not patched by Angular's zone. We fixed the issue using angular NgZones in the service when dealing with the Observers used to interact with google maps API's endpoints.

You can find more information about these issues here:

- https://gist.github.com/endash/1f961830d0c5b744598a
- http://stackoverflow.com/a/38100262/1116959

These are great posts that will help you understand NgZones:

- http://orizens.com/wp/topics/angular-2-ngzone-intro-the-new-scope-apply/
- http://blog.thoughtram.io/angular/2016/02/01/zones-in-angular-2.html

Icons

We also added some fancy stuff like the custom icons. According to google maps API there are two ways of customizing icons.

- Using images
- Using symbols (svg)

We followed the second option as we think is the best approach. If you want to change the icons symbol you need to find the path of the svg you want and set that value on the icon object. You can find more information here:

- http://lessons.livecode.com/m/33498/l/325452-getting-svg-path-data-for-svg-icon-extension
- https://developers.google.com/maps/documentation/javascript/symbols

And here you can find some handy svg icons you may find useful.

Distance between two points

We used Google distance matrix API to get the distance between two points when the user selects nearby places from the list.

Static Maps

For the static maps used in the contact card page, we used the Google Static Maps API. It's very easy to use and you will find all the information you need in the previous link.

Geolocation

This plugin provides information about the device's location, such as latitude and longitude. Learn more here.

To install run:

```
$ ionic plugin add cordova-plugin-geolocation --save

$ npm install --save @ionic-native/geolocation
```

Call Number

We use this plugin to call a number directly from the app.

<u>**To install run:**</u>

```
$ ionic plugin add call-number --save

$ npm install --save @ionic-native/call-number
```

<u>**Usage**</u>

The usage is very simple, we only need to know the phone number we want to call when the platform is ready.

TS

```
call(){
    this.platform.ready().then(() => {
      CallNumber.callNumber('555-555-555', true)
      .then(() => console.log('Launched dialer!'))
      .catch(() => console.log('Error launching dialer'));
    });
  }
```

The second parameter from the callNumber function is a boolean to bypass the app chooser. Set it to true if you always wish to bypass the app chooser if user has multiple applications installed that can handle calls.

HTML

```
<button ion-button clear (click)="call()">
   <ion-icon name="md-call"></ion-icon>
</button>
```

For more information go to https://github.com/Rohfosho/CordovaCallNumberPlugin

In App Browser

We use this plugin so users can view web pages without leaving your app.
You can show helpful articles, videos, and web resources inside of your app.

To install run:

```
$ ionic plugin add cordova-plugin-inappbrowser --save

$ npm install --save @ionic-native/in-app-browser
```

Usage

The implementation is very easy. We recommend to send as third parameter: `"location=yes"` in order to show a bar that allows the user to close the inAppBrowser and return to the app.

TS

```
openInAppBrowser(){
    this.platform.ready().then(() => {
      this.inAppBrowser('https://google.com', '_blank', "location=yes");
    })
  }
```

HTML

```
<a (click)=openInAppBrowser()>google.com</a>
```

For more information click here https://github.com/apache/cordova-plugin-inappbrowser

Send Email

This functionality allows users to send an email using one of the mail applications installed in their phone. Using Email Composer plugin this functionality can be implemented successfully.

To install run:

```
$ ionic plugin add --save cordova-plugin-email

$ npm install --save @ionic-native/email-composer
```

Usage example:

```
this.emailComposer.isAvailable().then((available: boolean) =>{
 if(available) {
   //Now we know we can send
 }
});

let email = {
  to: 'max@mustermann.de',
  cc: 'erika@mustermann.de',
  bcc: ['john@doe.com', 'jane@doe.com'],
  attachments: [
    'file://img/logo.png',
    'res://icon.png',
    'base64:icon.png//iVBORw0KGgoAAAANSUhEUg...',
    'file://README.pdf'
  ],
  subject: 'Cordova Icons',
  body: 'How are you? Nice greetings from Leipzig',
  isHtml: true
};

// Send a text message using default options
this.emailComposer.open(email);
```

TS

```
sendMail(){
    //for more option please go here: http://ionicframework.com/docs/native/email-composer/
     let email = {
      to: 'contact@ionicthemes.com',
      subject: 'This app is the best!',
      body: "Hello, I'm trying this fantastic app that will save me hours of development"
```

```
   };
   // Send a text message using default options
   this.emailComposer.open(email);
 }
```

HTML

```
 <button ion-button clear (click)="sendMail()">
       <ion-icon name="md-mail"></ion-icon>
  </button>
```

For more information visit the following page:
https://ionicframework.com/docs/native/email-composer/

Native Storage

NativeStorage allows you store to and retrieve data in the device. We use it in the app to store and retrieve the social logins (facebook and google) data.

To install run:

```
$ ionic plugin add --save cordova-plugin-nativestorage

$ npm install --save @ionic-native/native-storage
```

For more information please read: https://ionicframework.com/docs/native/native-storage/