

Catalogue Ionic 1.4 Documentation

Quick start Guide - Installation and user support

Created: Nov 25, 2015

Updated: Oct 25, 2016

By: AppSeed, <http://appseed.io/>

Thank you for purchasing our product. If you have any question that are beyond of the scope of this help file, please feel free to email via our [support center](#) form. Thank you!

Table of contents:

[Prerequisites](#)

[PhoneGap and Ionic](#)

[Tools](#)

[Run for the first time](#)

[Download and extract](#)

[Install libraries](#)

[Post installation](#)

[Linux/MacOX](#)

[Windows Users](#)

[Plugins](#)

[Run a local development server](#)

[Run in the emulator](#)

[Personalize the app](#)

[Keys](#)

[Configuration](#)

[News](#)

[Data Source configuration](#)

[Categories and Products](#)

[JSON Approach](#)

[Data Sources configuration](#)

[WooCommerce](#)

[Enable REST API](#)

[Generate API Keys](#)

[Set icons for product categories](#)

[Set the data provider](#)

[Wordpress](#)

[JSON API plugin](#)

[Drupal](#)
[Services and JSON View](#)
[Favorites](#)
[Email subject and body](#)
[Business Information](#)
[Push Notifications](#)
[Create Security Profiles](#)
[Plugins](#)
[Set Ionic Platform's credentials](#)
[Set GCM id](#)
[Send a Push Notification](#)
[POSTMAN](#)
[Ionic Platform UI](#)

[Support](#)
[References / Links:](#)
[Thank you](#)

Prerequisites

PhoneGap and Ionic

This is a [PhoneGap](#) and [Ionic](#) based application, so the [PhoneGap](#) and [Ionic](#) should be installed in your computer. Since Catalogue Ionic, targets iPhone and Android mobile devices, your environment should be properly configured and the corresponding SDK should be installed. If not, you will be still able to run the application into a Browser.

Please check the “[Install PhoneGap](#)” and “[Getting Started with Ionic](#)” sections in the official PhoneGap and Ionic sites respectively.

Ensure first that [NodeJS](#) and [GIT](#) is installed in your computer.

Tools

This project is based on the popular “[Ionic Framework Generator](#)” that boosts the overall development process by integrating a couple of very popular automation tools like [Grunt](#) and [Bower](#).

Install these tools by following the instructions in their corresponding web pages:

1. [Install Bower](#)
2. [Getting started with Grunt - Install the CLI](#)
3. [Getting started with Yeoman](#)

Finally install the yeoman generator via:

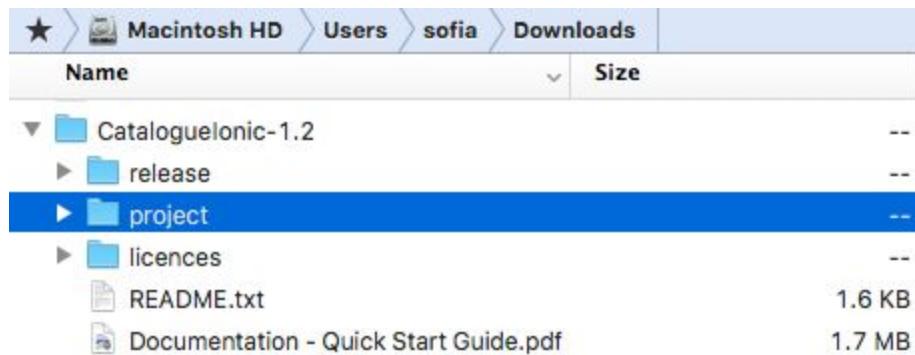
```
$ npm install -g generator-ionic
```

Run for the first time

In the screen captures that follow, we will demonstrate the process of preparing your environment and running the project for the first time.

Download and extract

Download the provided .zip file and extract it, you will see something similar to what is shown in the next screen:



The highlighted folder is the Ionic Project's directory.

Install libraries

Open a terminal window and navigate to the directory where the “catalogue” folder is located.

Install NodeJS dependences:

```
$ npm install
themess-mini:~ sofia$ catalogue — bash — 80x5
themess-mini:~ sofia$ themess-mini:~ sofia$ themess-mini:~ sofia$ cd Documents/catalogue
themess-mini:catalogue sofia$ npm install
```

Post installation

There is a post installation process under which required Cordova plugins and Javascript dependencies are installed. To simplify this process two scripts are already prepared for both platforms: Linux/MacOS and Windows

Linux/MacOX

Install all the required plugins and Javascript dependencies:

```
$ ./install.sh
```

Windows Users

Similarly, Windows users should execute:

```
$ install.bat
```

Plugins

Build your project for the first time. This will create the [www] folder which is the actual cordova directory and where the plugins will be installed.

```
$ grunt build
```

Follow the same process as with “Libraries” and install the required plug ins by using the commands that follow:

```
$ cordova plugin add {plugin name or url}
```

eg:

```
$ cordova plugin add cordova-plugin-inappbrowser
```

Please check the `Readme.md` file in the project folder for the list with all the required plugins the application is based on and should be installed.

Run a local development server

Start the local NodeJS server and run the application in the browser:

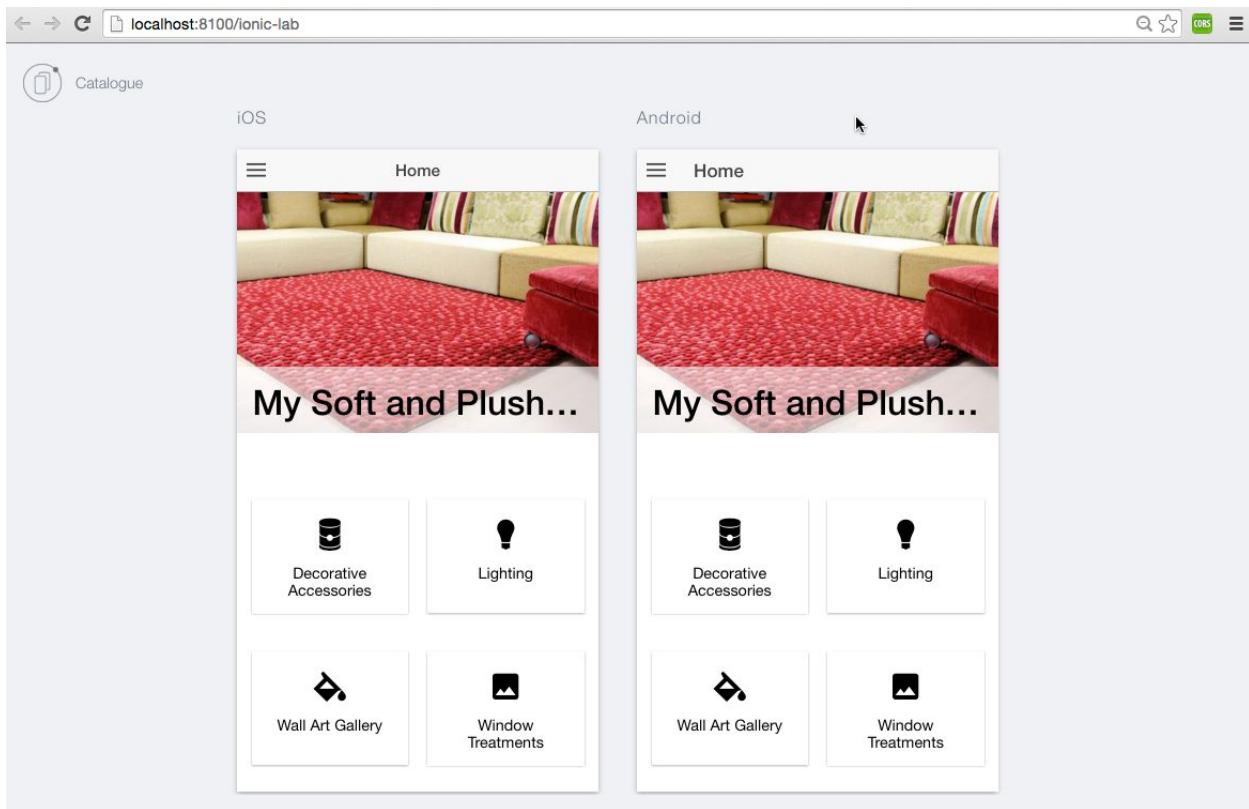
```
$ grunt serve --lab
```



A screenshot of a terminal window titled "catalogue – bash – 80x5". The window shows the following command history and output:

```
themess-mini:~ sofia$  
themess-mini:~ sofia$  
themess-mini:~ sofia$  
themess-mini:~ sofia$ cd Documents/catalogue  
themess-mini:catalogue sofia$ grunt serve --lab
```

A browser window will open with two virtual devices the one next to the other.



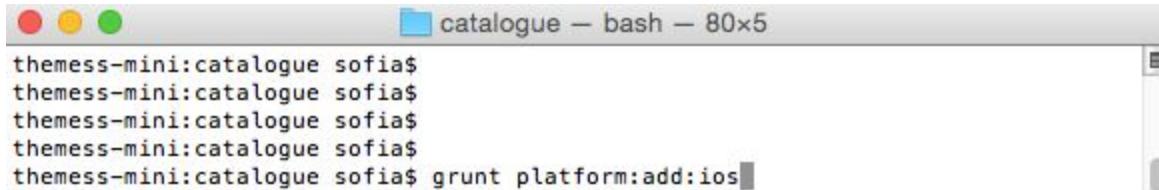
You could also open the application in a single browser window by starting it with the command:

```
$ grunt serve
```

Run in the emulator

First the preferred platform should be added. In this case iOS:

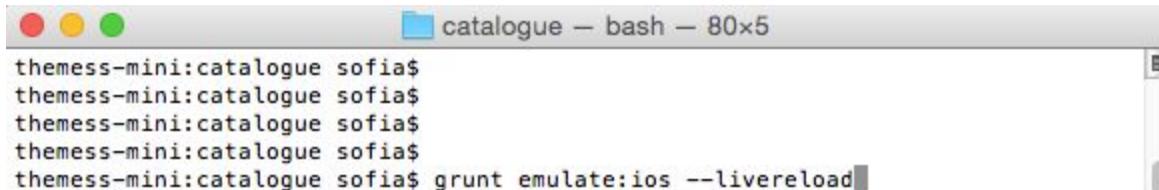
```
$ grunt platform:add:ios
```



```
catalogue — bash — 80x5
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ grunt platform:add:ios
```

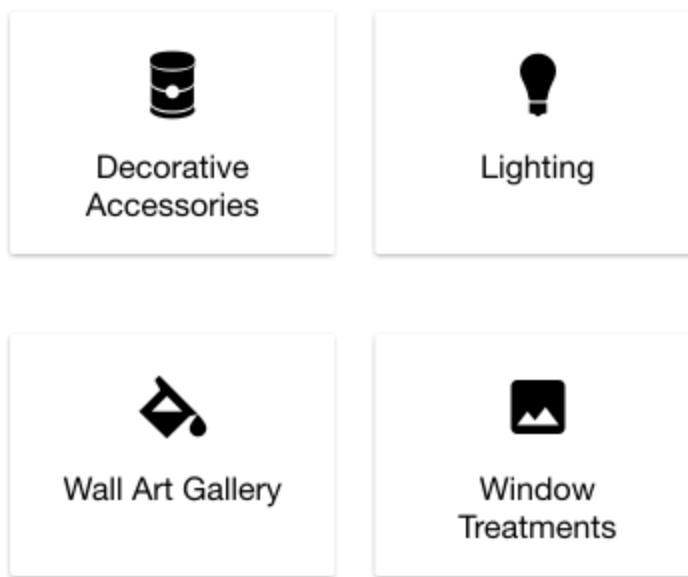
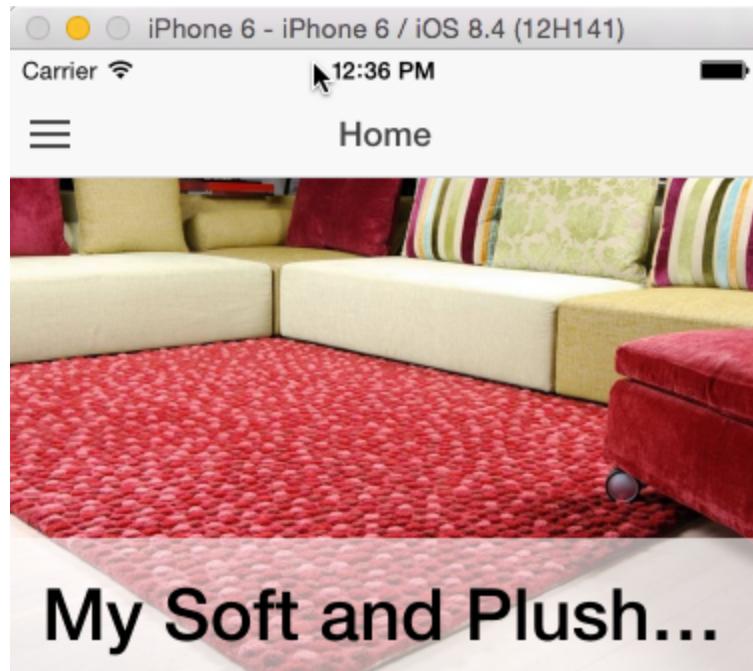
Now the application is ready to start inside a simulator:

```
grunt emulate:ios --livereload
```



```
catalogue — bash — 80x5
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ 
themess-mini:catalogue sofia$ grunt emulate:ios --livereload
```

The iPhone simulator will launch and the Catalogue Ionic app will start.



Personalize the app

The first step, once you get familiar with the application, is to personalize it. In order to do this, edit the `ionic.project` and `config.xml` files and replace the highlighted fields:

```

 ionic.project

1 {
2   "name": "catalogue",
3   "app_id": "c52a7655"
4 }

```

ionic.project

```

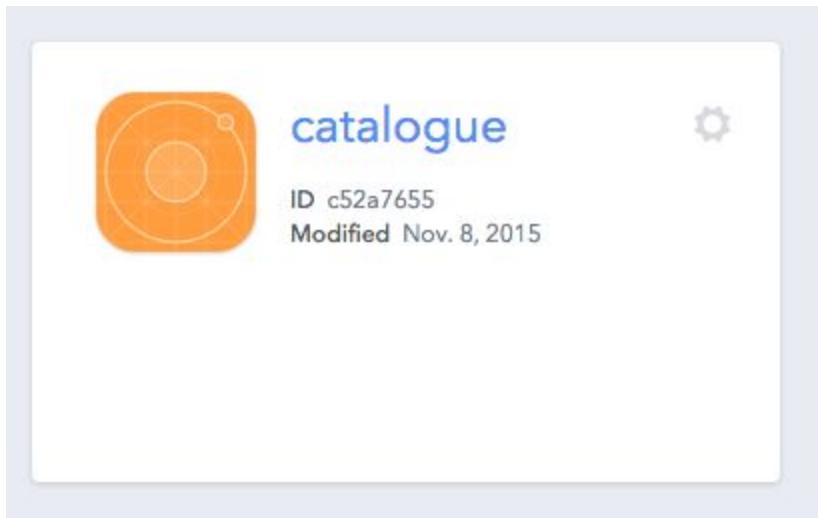
config.xml

1 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
2 <widget id="com.titaniumtemplates.catalogue" version="0.0.1" xmlns="http://www.w3.org/ns/wid
3   <name>Catalogue</name>
4   <description>
5     Catalogue. The bootstrap you need in order to build your next Ionic application.
6   </description>
7   <author email="skounis@gmail.com" href="http://about.me/stavros.kounis">
8     Stavros Kounis
9   </author>
10  <content src="index.html"/>

```

config.xml

As for the app_id, use the ID of the corresponding application in your ionic.io account.



Keys

Create a set of Public and Secret key for this app in your ionic.io account.

The screenshot shows the Ionic.io application interface. The top navigation bar includes a back arrow, a refresh icon, and the URL <https://apps.ionic.io/app/c52a7655/config/keys>. On the right side of the header are icons for shield, star, and CORS. Below the header, there's a navigation menu with a 'catalogue' item and a dropdown arrow, followed by 'Settings'. The main content area has a title 'API Keys' with a subtitle explaining that API keys allow communication with services. It lists existing keys: 'Public Key' (Master API Key) and 'Secret Key' (with a 'Show secret' link). A 'New API Key' button is at the bottom. The left sidebar contains icons for Push, Analytics, Deploy, Users, and Settings, with 'Settings' being the active tab.

API Keys

API keys allow your application to communicate with the ionic.io services. You can create multiple API keys that have different permissions to restrict the actions your apps can perform.

KEY	DESCRIPTION	DELETE
Public Key	Master API Key	DELETE
Secret Key	Show secret	

[New API Key](#)

Use these keys and configure the related properties in the Gruntfile.js

Configuration

News

News screen is configured to fetch data from a remote source. JSON protocol is used for this purpose.

The remote data source that is already configured for demonstrating purposes is available in the URL below:

- <http://skounis.s3.amazonaws.com/mobile-apps/local-business/news.json>

Data Source configuration

In order to set the **Catalogue Ionic** application to work with your own news data source you should follow the next steps:

1. Create your JSON structures and put them online.
2. Open the related `news.service.js` file, locate the `url` variable and replace it with your source.



```
scripts
  catalogs
  categories
  common
  contact-us
  drupal
  favorites
  home
  map
  menu
  news
    article.controller.js
    article.html
    articles.controller.js
    articles.html
    news.module.js
    news.service.js
  products
  push

news.service.js

1 (function() {
2   'use strict';
3
4   angular
5     .module('catalogue.news')
6     .factory('newsService', newsService);
7
8   newsService.$inject = ['$http', '$q'];
9
10  /* @ngInject */
11  function newsService($http, $q) {
12    var url = 'http://skounis.s3.amazonaws.com/mobile-apps/local-business/news.json';
13    var result = [];
14
15    var service = {
16      all: all,
17      get: get
18    };
19
20    return service;
21  }
22
23  /* @ngInject */
24  function all($q) {
25    var def = $q.all();
26
27    def.resolve(result);
28
29    return def;
30  }
31
32  /* @ngInject */
33  function get($q, id) {
34    var def = $q.when(result[id]);
35
36    def.resolve(def);
37
38    return def;
39  }
40
41  /* @ngInject */
42  function newsService($http, $q) {
43    var url = 'http://skounis.s3.amazonaws.com/mobile-apps/local-business/news.json';
44    var result = [];
45
46    var service = {
47      all: all,
48      get: get
49    };
50
51    return service;
52  }
53
54  /* @ngInject */
55  function all($q) {
56    var def = $q.all();
57
58    def.resolve(result);
59
60    return def;
61  }
62
63  /* @ngInject */
64  function get($q, id) {
65    var def = $q.when(result[id]);
66
67    def.resolve(def);
68
69    return def;
70  }
71
72  /* @ngInject */
73  function newsService($http, $q) {
74    var url = 'http://skounis.s3.amazonaws.com/mobile-apps/local-business/news.json';
75    var result = [];
76
77    var service = {
78      all: all,
79      get: get
80    };
81
82    return service;
83  }
84
85  /* @ngInject */
86  function all($q) {
87    var def = $q.all();
88
89    def.resolve(result);
90
91    return def;
92  }
93
94  /* @ngInject */
95  function get($q, id) {
96    var def = $q.when(result[id]);
97
98    def.resolve(def);
99
100   return def;
101 }

news.service.js
```

Categories and Products

The application can consume JSON files that you have created beforehand and provided to the application uploading them online (e.g. Amazon S3). Alternatively, in case you have a [WooCommerce](#) website, you can connect the application to consume data that already exist on your site.

JSON Approach

Categories are fetched from a remote JSON file consisting of all the available categories of products, each one of them in the form of a URL where the JSON source is located. As an example, a categories JSON file is already configured and available in the URL below:

- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/categories.json>

Also, if a category has the “featured” property set as “true”, it will be displayed on the home screen. The default value of this property is “false”.

Therefore, you will need to construct a JSON file with all the URLs pointing to each category and a JSON file of the category itself. For demonstrating purposes, there are six categories, namely six more JSON files. The JSON sources for all the categories used as an example are the following:

- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-a.json>
- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-b.json>
- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-c.json>
- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-d.json>
- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-e.json>
- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/cat-f.json>

Furthermore, at the very top of the home screen, there is a slide box where some of the products on sale or on promotion campaigns could be displayed. For that reason, a separate JSON file containing all those products is used:

- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/featured.json>

Finally, all the business related information such as address, email, name etc are fed by the following JSON file:

- <http://skounis.s3.amazonaws.com/mobile-apps/catalogue/business.json>

Data Sources configuration

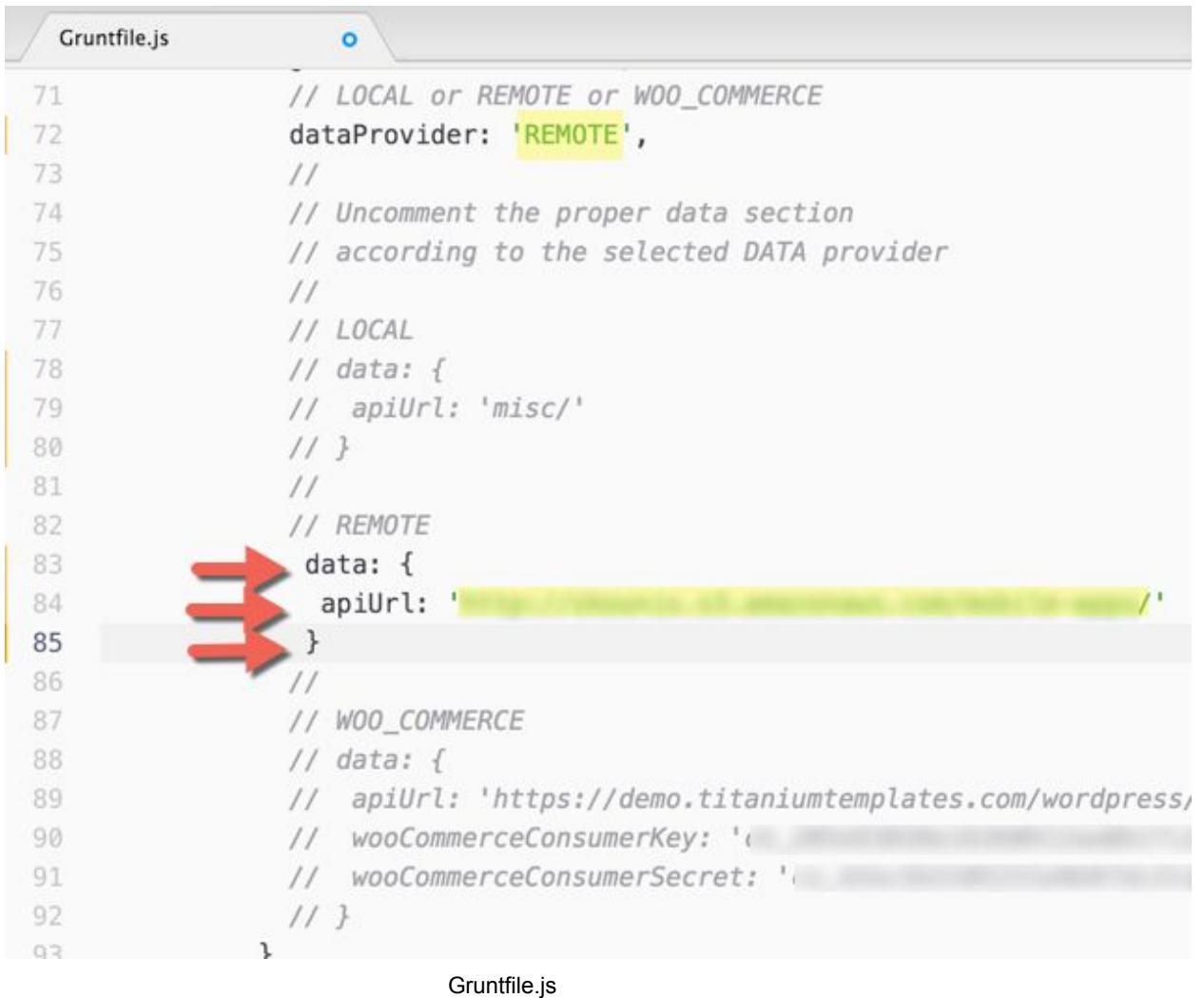
In order to set the categories data source, you should follow the next steps:

1. Create the JSON structure of each category and put it online.
2. Create the JSON structure containing the URLs of all the categories and put it online.
3. Create the JSON structure of the products that will be displayed on the home screen and put it online.
4. Create the JSON structure of the business related information and put it online.
5. Set the data provider:
 - a. In `Gruntfile.js`, set the data provider to *LOCAL* if the data is pulled from the `local/misc` folder and make sure the related lines are uncommented:

```
Gruntfile.js
```

```
95      },
96      production: {
97        constants: {
98          ENV: {
99            name: 'production',
100           youtubeKey: 'AIzaSyDael5MmCQa1GKQNKQYypmBeB08GATgSEo',
101           ionicPrivateKey: 'a9265eaf15a20cc8516c770e8748aeed4891b28f453ce755',
102           //'c63b22cd7330a4a9d5e526bfcd74891a59fd5c23d1d81239',
103           ionicPublicKey: 'e30d4d540b8c75d1f167bbf242423c3fb23fe10275d1c016',
104           //'04dee953a91ef3857f1c8a7cf4748ecee375848681f7e833',
105           ionicAppId: '241b6d37', //'2113c758',
106           gcmId: '228071472080',
107           // LOCAL or REMOTE or WOO_COMMERCE
108           dataProvider: 'LOCAL',
109           //
110           // Uncomment the proper data section
111           // according to the selected DATA provider
112           //
113           // LOCAL
114           data: {
115             apiUrl: 'misc/'
116           }
117           //
118           // REMOTE
119           // data: {
```

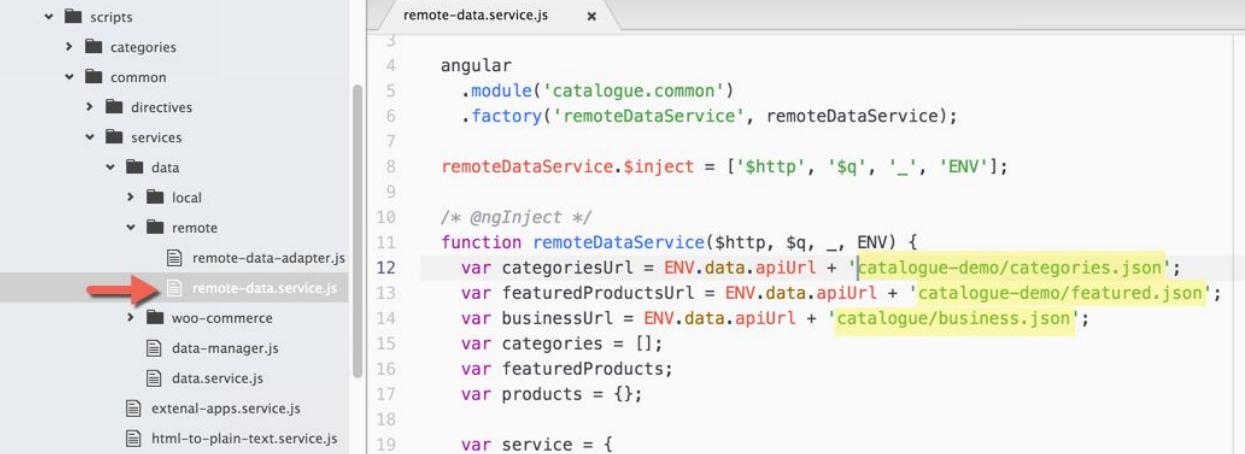
- b. In case that the data is pulled from a remote source, set the data provider to **REMOTE** and the related `apiUrl`, as shown below:



```
Gruntfile.js
71      // LOCAL or REMOTE or WOO_COMMERCE
72      dataProvider: 'REMOTE',
73      //
74      // Uncomment the proper data section
75      // according to the selected DATA provider
76      //
77      // LOCAL
78      // data: {
79      //   apiUrl: 'misc/'
80      // }
81      //
82      // REMOTE
83      data: {
84        apiUrl: 'https://REDACTED.amazonaws.com/wordpress'
85      }
86      //
87      // WOO_COMMERCE
88      // data: {
89      //   apiUrl: 'https://demo.titaniumtemplates.com/wordpress',
90      //   woocommerceConsumerKey: '',
91      //   woocommerceConsumerSecret: ''
92      // }
```

Gruntfile.js

Note that the `apiUrl` points to the S3 bucket's path where the JSON files are located. Additionally, open the `remote-data.service.js` file under the `app/scripts/common/services/data` path, locate the `categoriesUrl`, `featuredProductsUrl` and `businessUrl` variables and replace them with the sources created in step 2, 3 and 4 respectively. Note that the url should point to the exact JSON file.



```

3
4 angular
5   .module('catalogue.common')
6   .factory('remoteDataService', remoteDataService);
7
8   remoteDataService.$inject = ['$http', '$q', '_', 'ENV'];
9
10  /* @ngInject */
11  function remoteDataService($http, $q, _, ENV) {
12    var categoriesUrl = ENV.data.apiUrl + 'catalogue-demo/categories.json';
13    var featuredProductsUrl = ENV.data.apiUrl + 'catalogue-demo/featured.json';
14    var businessUrl = ENV.data.apiUrl + 'catalogue/business.json';
15    var categories = [];
16    var featuredProducts;
17    var products = {};
18
19    var service = {

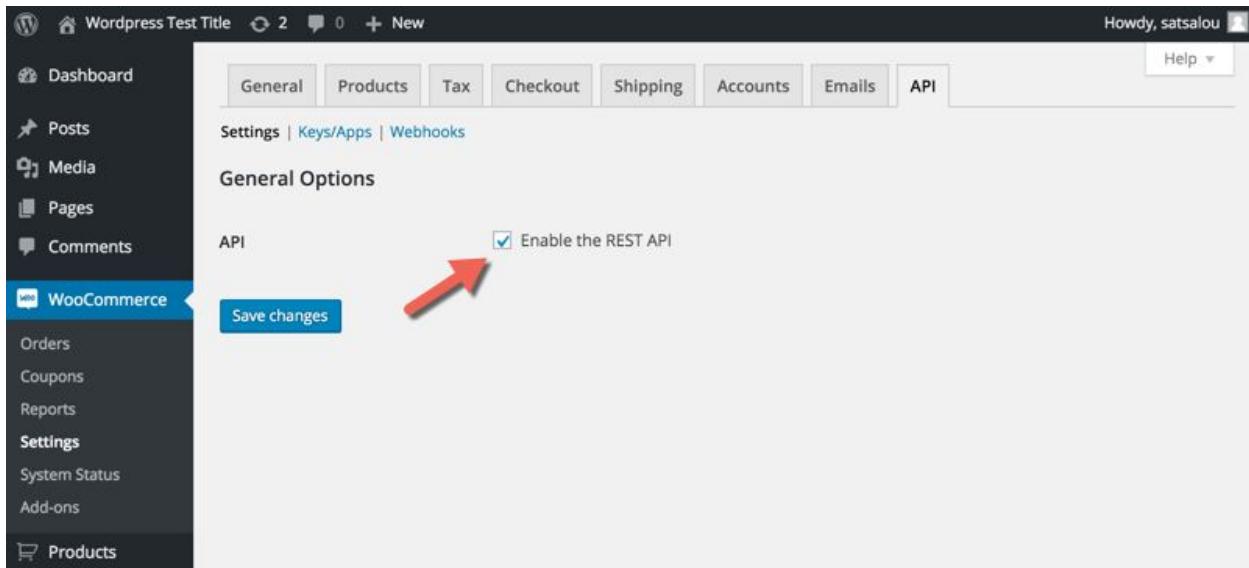
```

remote-data.service.js

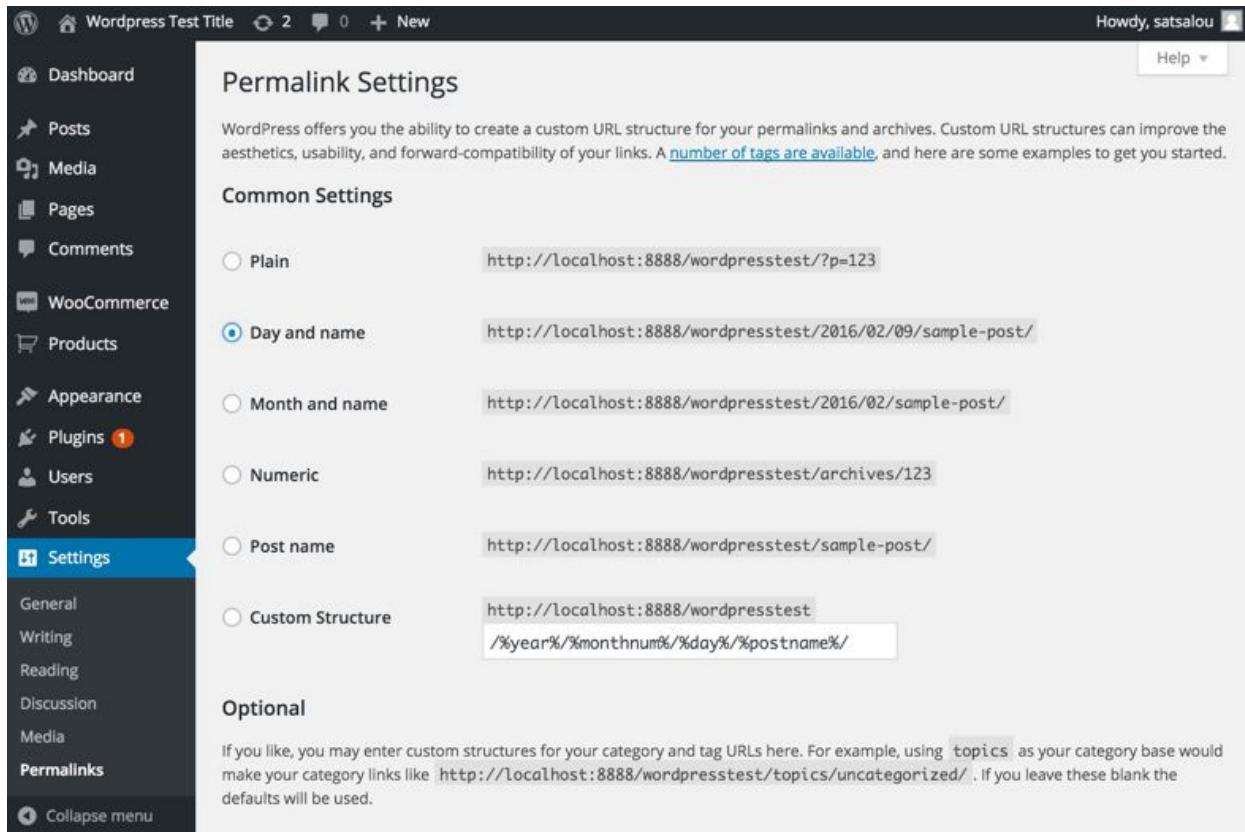
WooCommerce

Enable REST API

Firstly, you should enable the WooCommerce REST API. In order to do that, visit **WooCommerce > Settings > API** tab and tick the “Enable REST API” checkbox.



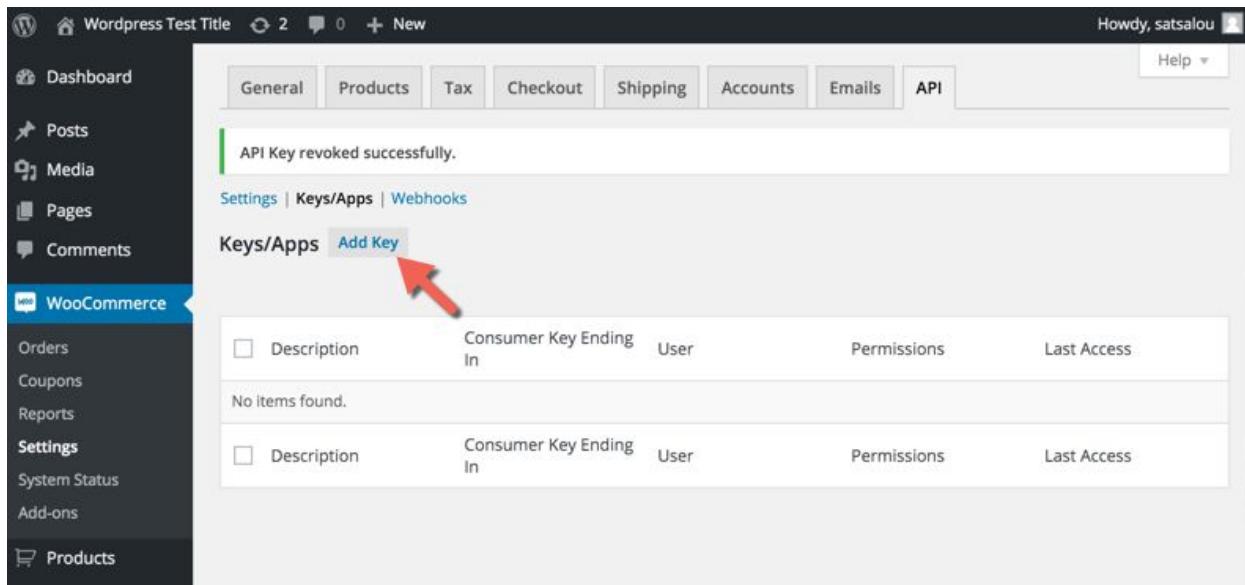
Please, note that you should have turned on the Wordpress Permalinks (**Settings > Permalinks**).



The screenshot shows the WordPress Permalink Settings page. The left sidebar is dark grey with white text, showing navigation links like Dashboard, Posts, Media, Pages, Comments, WooCommerce, Products, Appearance, Plugins (with 1 notification), Users, Tools, and Settings. The 'Permalinks' link under Settings is highlighted with a blue background. The main content area has a light grey background. At the top, it says 'Permalink Settings'. Below that, a note states: 'WordPress offers you the ability to create a custom URL structure for your permalinks and archives. Custom URL structures can improve the aesthetics, usability, and forward-compatibility of your links. A [number of tags are available](#), and here are some examples to get you started.' Under 'Common Settings', there are five radio button options: 'Plain' (localhost:8888/wordpress/test/?p=123), 'Day and name' (localhost:8888/wordpress/test/2016/02/09/sample-post/), 'Month and name' (localhost:8888/wordpress/test/2016/02/sample-post/), 'Numeric' (localhost:8888/wordpress/test/archives/123), 'Post name' (localhost:8888/wordpress/test/sample-post/), and 'Custom Structure' (localhost:8888/wordpress/test/%year%/%monthnum%/%day%/%postname%/). The 'Custom Structure' input field contains the placeholder '/%year%/%monthnum%/%day%/%postname%/'. Below this, under 'Optional', there is a note about creating custom category and tag URLs.

Generate API Keys

To create the keys, go to WooCommerce > Settings > API > Keys/Apps.



The screenshot shows the WooCommerce API Keys/Apps page. The left sidebar is dark grey with white text, showing navigation links for WooCommerce: Orders, Coupons, Reports, Settings (System Status, Add-ons), and Products. The 'Settings' link under WooCommerce is highlighted with a blue background. The main content area has a light grey background. At the top, it shows 'General', 'Products', 'Tax', 'Checkout', 'Shipping', 'Accounts', 'Emails', and 'API' tabs, with 'API' selected. A green message box says 'API Key revoked successfully.' Below that, there are three tabs: 'Settings', 'Keys/Apps' (which is active and highlighted with a blue background), and 'Webhooks'. A red arrow points to the 'Add Key' button, which is highlighted with a blue background. The table below shows two rows of API keys:

Description	Consumer Key Ending In	User	Permissions	Last Access
No items found.				

Take a note of the “Consumer Key” and the “Consumer Secret” as they will be used later.

The screenshot shows the WordPress admin interface for a site titled "Wordpress Test Title". The left sidebar has a "WooCommerce" section selected. The main content area is titled "Key Details" and displays two generated API keys: "Consumer Key" and "Consumer Secret". A QR code is also provided. A green message box at the top says: "API Key generated successfully. Make sure to copy your new API keys now. You won't be able to see it again!". Red arrows point from the text "Set icons for product categories" below to the "Consumer Key" and "Consumer Secret" fields.

Set icons for product categories

In order to set the icons for each product category showing on Home screen and in menu, you should declare the product category ID and then the icon you wish to be displayed for this category. This is taking place in `woo-commerce-data-adapter.js` file under `scripts/common/services/data/woo-commerce/` path:

```

1  (function () {
2    'use strict';
3
4    angular
5      .module('catalogue.common')
6      .factory('woocommerceDataAdapter', woocommerceDataAdapter);
7
8    woocommerceDataAdapter.$inject = ['$'];
9
10   /* @ngInject */
11   function woocommerceDataAdapter(_) {
12
13     var iconsDictionary = {
14       '20': 'icon ion-tshirt',
15       '21': 'icon ion-male',
16       '22': 'icon ion-music-note',
17       '23': 'icon ion-aperture',
18       '24': 'icon ion-ipod',
19       '25': 'icon ion-tshirt',
20       '26': 'icon ion-headphone',
21       'default': 'ion-android-folder',
22     };
23

```

woo-commerce-data-adapter.js

Note that you can find the category ID under the “Products” tab by hovering on a particular product:

	Name	SKU	Stock	Price	Categories	Tags	★	🛒	Date
<input type="checkbox"/>	Demo Product ID: 10 Edit Quick Edit Trash View Duplicate	-	In stock	-	-	-			Published 1 min ago
<input type="checkbox"/>	Name	SKU	Stock	Price	Categories	Tags	★	🛒	Date

Also, notice that you can set a default icon in case that you want to show a default icon for some product categories which do not have a defined icon.

Set the data provider

In `Gruntfile.js` file, set the data provider to `WOOCOMMERCE`, uncomment the indicated lines and replace the variables with your own api url and the keys mentioned earlier:



```
Gruntfile.js
72    dataProvider: 'WOO_COMMERCER',
73     //
74     // Uncomment the proper data section
75     // according to the selected DATA provider
76     //
77     // LOCAL
78     // data: {
79     //   apiUrl: 'misc/'
80     // }
81     //
82     // REMOTE
83     // data: {
84     //   apiUrl: 'http://skounis.s3.amazonaws.com/mobile-apps/'
85     // }
86     //
87     // WOO_COMMERCER
88     data: {
89       apiUrl: 'https://demo.titaniumtemplates.com/wordpress/wc-api/v3/',
90       woocommerceConsumerKey: '████████████████████████████████████████',
91       woocommerceConsumerSecret: '████████████████████████████████████████'
92     }
93 }
```

Gruntfile.js

Note that your api url should consist of your home url (in this example <https://demo.titaniumtemplates.com/wordpress/>) and followed by `wc-api/` and the [version](#) of the WooCommerce API you use (in this example `v3/`).

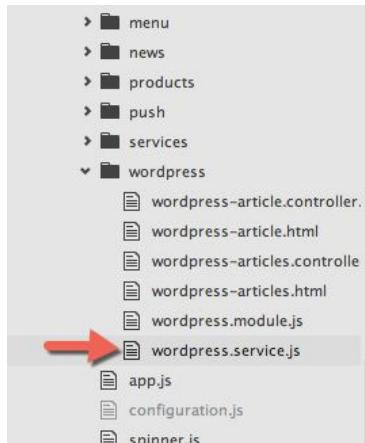
Wordpress

This feature loads articles from a remote Wordpress site. [Wordpress JSON API](#) is used for the creation of JSON feed of the posts.

For demonstration purposes a Wordpress website has been installed. Its URL and JSON feed that is used in this application are provided by the following links:

- Site: <http://demo.titaniumtemplates.com/wordpress/>
- JSON: <http://demo.titaniumtemplates.com/wordpress/?json=1>.

To replace this JSON source with yours, open the `wordpress.service.js` file located under the `app/scripts/wordpress/` path and place your link where the arrow shows.



```

menu
news
products
push
services
wordpress
  wordpress-article.controller.js
  wordpress-article.html
  wordpress-articles.controller.js
  wordpress-articles.html
  wordpress.module.js
  wordpress.service.js (highlighted by a red arrow)
  app.js
  configuration.js
  spinner.js

```

```

1 (function() {
2   'use strict';
3
4   angular
5     .module('catalogue.wordpress')
6     .factory('wordpressService', wordpressService);
7
8   wordpressService.$inject = ['$http', '$q', '_', 'htmlToPlainText'];
9
10 /* @ngInject */
11 function wordpressService($http, $q, _, htmlToPlainText) {
12   var url = 'http://demo.titaniumtemplates.com/wordpress/?json=1';
13   var articles = [];
14

```

wordpress.service.js

JSON API plugin

Please note that, firstly, the [JSON API plugin](#) needs to be installed and activated. The related [documentation](#) is also available. According to that, the generation of the feed is done by finding the location on a website that you want to get a JSON feed and add “?json=1” at the end.

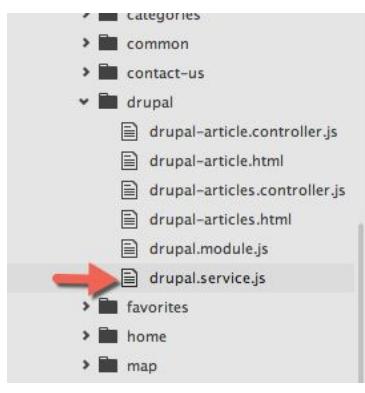
Drupal

This feature loads articles from a remote Drupal web site. [Services](#) module is used for the creation of JSON feed of the posts.

For demonstration purposes a Drupal website has been installed. Its URL and JSON feed that is used in this application are provided by the following links:

- Site: <http://demo.titaniumtemplates.com/drupal>
- JSON: http://demo.titaniumtemplates.com/drupal/rest/views/rest_api.

To replace this JSON source with yours, open the `drupal.service.js` file located under the `app/scripts/drupal/` path and place your link where the arrow shows.



```

categories
common
contact-us
drupal
  drupal-article.controller.js
  drupal-article.html
  drupal-articles.controller.js
  drupal-articles.html
  drupal.module.js
  drupal.service.js (highlighted by a red arrow)
  favorites
  home
  map

```

```

1 (function() {
2   'use strict';
3
4   angular
5     .module('catalogue.drupal')
6     .factory('drupalService', drupalService);
7
8   drupalService.$inject = ['$http', '$q', '_', 'htmlToPlainText'];
9
10 /* @ngInject */
11 function drupalService($http, $q, _, htmlToPlainText) {
12   var url = 'http://demo.titaniumtemplates.com/drupal/rest/views/rest_api';
13   var articles = [];
14
15   var service = {
16     getArticles: getArticles,

```

drupal.service.js

Services and JSON View

Using Drupal, initially, you should install the [Services](#) module and enable it. A View that exposes a JSON feed of the articles should also be created.

The following tutorial describes this process:

- A Beginners Guide to the Drupal Services Module
<https://www.ostraining.com/blog/drupal/services/>

All the modules that are needed for this are listed below:

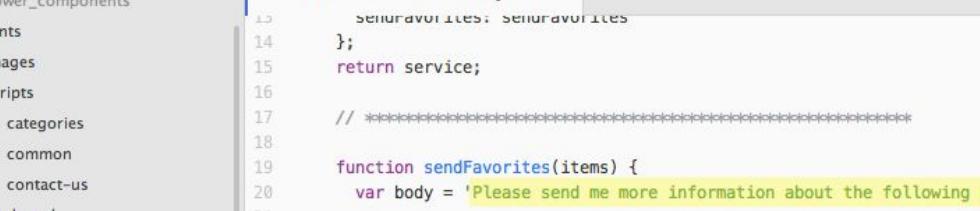
1. <https://www.drupal.org/project/services>
 2. <https://www.drupal.org/project/ctools>
 3. <https://www.drupal.org/project/libraries>
 4. <https://www.drupal.org/project/views>
 5. https://www.drupal.org/project/services_views

Favorites

On Favorites screen, the user is able to send their favorites products to an email address. For example, a reason to do this could be asking for more information about the products.

Email subject and body

In `favorites-sender.service.js` file located under the `app/scripts/favorites/` path, you can edit the **body** and the **subject** of the email as shown below:



The screenshot shows the Atom code editor interface. The left sidebar displays the project structure:

- app
 - bower_components
 - fonts
 - images
 - scripts
 - categories
 - common
 - contact-us
 - drupal
 - favorites**
 - home
 - map
 - menu
 - newsA red arrow points to the file `favorites-sender.service.js` in the `favorites` folder.

Business Information

All the business information such as store name, address, description, email, office location, phone number, social networks and map annotations can be configured in the `business.json` file and stored either locally in the `misc` folder or online, for example Amazon S3. The structure of the JSON file should look similar to the following:

```
business.json
```

```

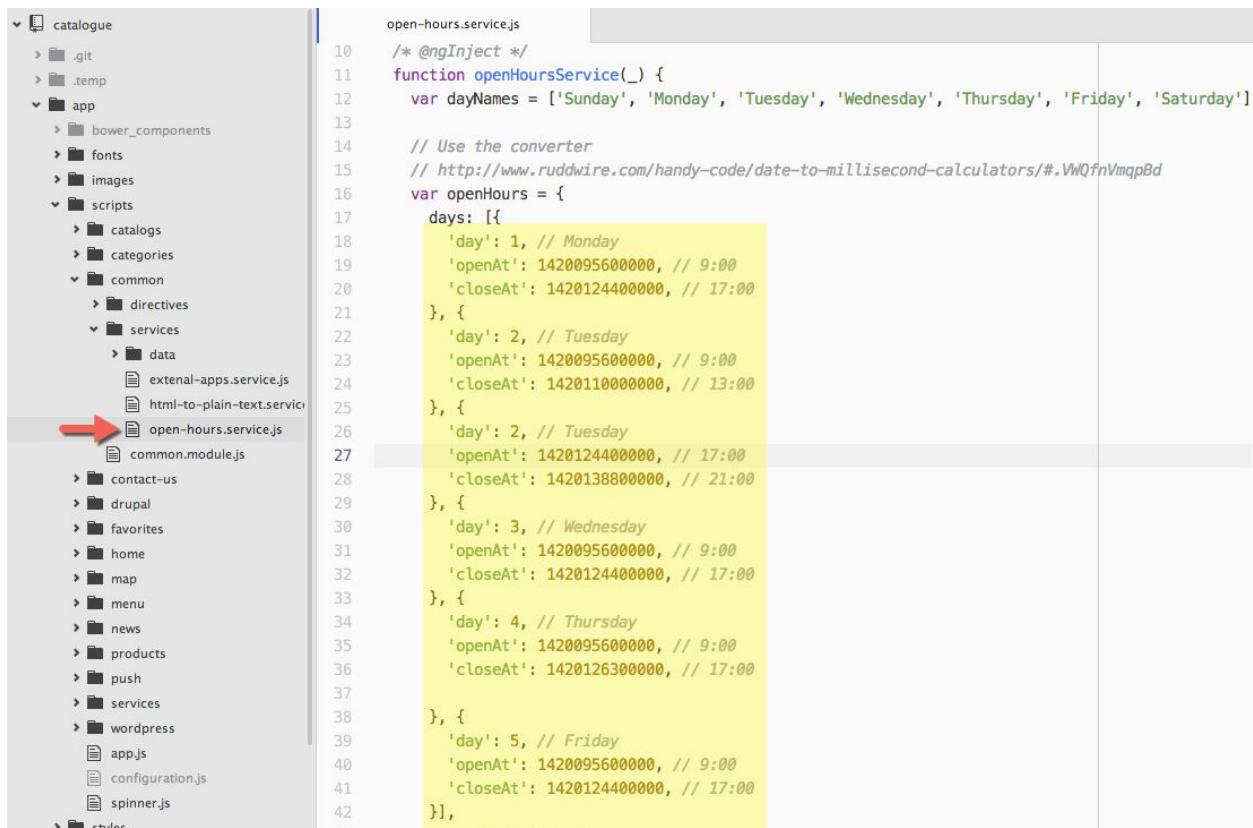
1  {
2      "id": 1,
3      "jsonrpc": "2.0",
4      "result": {
5          "storeName": "Home Decoration",
6          "address": "50 Market Street, San Francisco, California 94103, United States",
7          "desc": "Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod
8          "phoneNumber": "+306973216110",
9          "email": "skounis@gmail.com",
10         "officeLocation": "37.7736854,-122.421034",
11         "facebookPage": "https://www.facebook.com/ionicframework",
12         "instagramPage": "https://instagram.com/phonegap/",
13         "twitterPage": "https://twitter.com/titemplates/",
14         "pinterestPage": "https://www.pinterest.com/",
15         "map": {
16             "origin": {
17                 "latitude": 37.407,
18                 "longitude": -122.1
19             },
20             "zoomLevel": 15,
21             "annotations": [
22                 {
23                     "title": "Molestie et wisi.",
24                     "latitude": 37.407,
25                     "longitude": -122.1
26                 },
27                 {
28                     "title": "Ullamcorper eros.",
29                     "latitude": 37.41,
30                     "longitude": -122.1
31                 }
32             ]
33         }
34     }
35 }
```

business.json

Note that the `origin` property of map annotations sets the point where the map will be

centered. You are able to set as many annotation points as you wish. To do this, you need to extend the existing JSON structure.

In order to set the hours at which the company is open, open the `open-hours.service.js` file located under the `app/scripts/common/services` path and adjust the `openHours` array. The open hours are going to be displayed on the `Contact us` screen too. Note that the hours should be represented in milliseconds since 1970/01/01 in GMT timezone.



```
/* @ngInject */
function openHoursService(_) {
  var dayNames = ['Sunday', 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday']

  // Use the converter
  // http://www.ruddwire.com/handy-code/date-to-millisecond-calculators/#.VWQfnVmqpBd
  var openHours = {
    days: [
      {day: 1, // Monday
       'openAt': 142009560000, // 9:00
       'closeAt': 142012440000, // 17:00},
      {day: 2, // Tuesday
       'openAt': 142009560000, // 9:00
       'closeAt': 142011000000, // 13:00},
      {day: 3, // Wednesday
       'openAt': 142009560000, // 9:00
       'closeAt': 142012440000, // 17:00},
      {day: 4, // Thursday
       'openAt': 142009560000, // 9:00
       'closeAt': 142012630000, // 17:00},
      {day: 5, // Friday
       'openAt': 142009560000, // 9:00
       'closeAt': 142012440000, // 17:00}
    ],
    zone: 3 // GMT + 3
  }
}
```

open-hours.service.js

Push Notifications

Before you proceed further please check the related documentation provided by Ionic:

<https://docs.ionic.io/services/push/>

Create Security Profiles

Firstly, you should create a security profile with push credentials for Android. For more information, please, visit [Ionic official documentation](#). Similarly, you should create a security profile for iOS as the [Ionic documentation](#) describes.

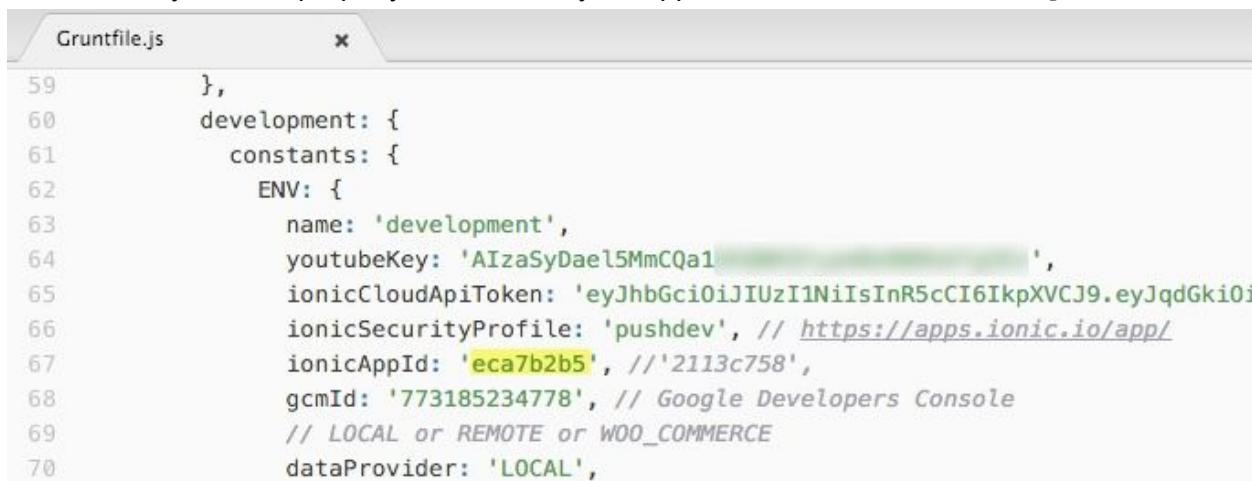
Plugins

Install the required plugins by using the following command where SENDER_ID is your GCM project number:

```
$ cordova plugin add phonegap-plugin-push --variable SENDER_ID=12341234
```

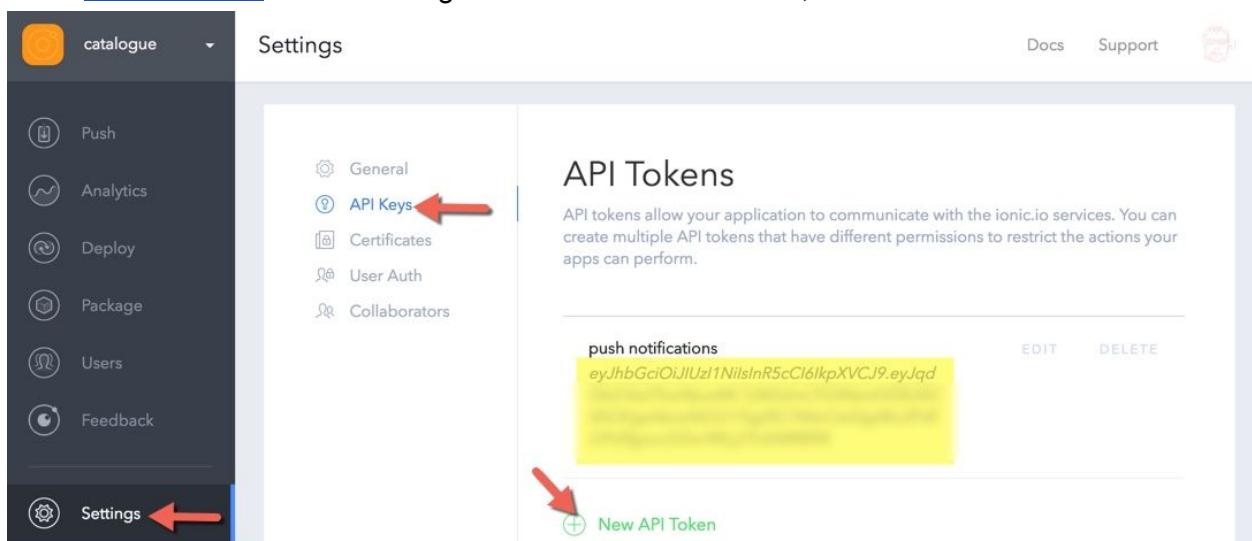
Set Ionic Platform's credentials

Please, confirm that you have already created the corresponding Application in your ionic.io account and you have properly set the ID of your application in the `Gruntfile.js`



```
Gruntfile.js
59      },
60      development: {
61        constants: {
62          ENV: {
63            name: 'development',
64            youtubeKey: 'AIzaSyDaeI5MmCQa1',
65            ionicCloudApiKey: 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOi',
66            ionicSecurityProfile: 'pushdev', // https://apps.ionic.io/app/
67            ionicAppId: 'eca7b2b5', // '2113c758',
68            gcmId: '773185234778', // Google Developers Console
69            // LOCAL or REMOTE or WOO_COMMERCE
70            dataProvider: 'LOCAL',
```

Also, you will need an API token so you can use Ionic API to send push notifications. Simply, go to the [Ionic Platform](#) and hit settings to create a token. Please, make a note of this token:



The screenshot shows the Ionic Platform interface. On the left, there's a sidebar with icons for Push, Analytics, Deploy, Package, Users, Feedback, and Settings. A red arrow points to the 'Settings' icon. The main content area is titled 'API Tokens'. It has a sidebar with 'General', 'API Keys' (which is highlighted with a red arrow), 'Certificates', 'User Auth', and 'Collaborators'. Below this, there's a table for 'push notifications' with one entry: 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOi...'. A red arrow points to the 'New API Token' button at the bottom right of this table.

Additionally, define your iOS and Android security profiles by creating a certificate on the [Ionic](#)

[Platform](#). Please, make a note of your certificate name:

The screenshot shows the Ionic Cloud Settings interface. On the left, there's a sidebar with icons for Push, Analytics, Deploy, Package, Users, Feedback, and Settings. The Settings icon is highlighted with a red arrow. The main area is titled "Security Profiles & Credentials". It contains a sub-header: "Security Profiles group your app credentials into a single entity which is used in the Ionic Platform by various services. Learn more about Security Profiles in our [docs](#)." Below this, there's a table with two columns: "NAME" and "TAG". A single row is listed: "pushdev" under NAME and "pushdev" under TAG. There are "EDIT" and "DELETE" buttons next to the row. At the bottom left of the main area, there's a green button labeled "+ New Security Profile" with a plus sign icon.

Finally, set your `ionicCloudApiKey` and `ionicSecurityProfile` with your API token and certificate name in `Gruntfile.js`:

```
Gruntfile.js
59      },
60      development: {
61        constants: {
62          ENV: {
63            name: 'development',
64            youtubeKey: 'AIzaSyDaeI5MmCQa1',
65            ionicCloudApiKey: 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOi
66            ionicSecurityProfile: 'pushdev', // https://apps.ionic.io/app/
67            ionicAppId: 'eca7b2b5', // '2113c758',
68            gcmId: '773185234778', // Google Developers Console
69            // LOCAL or REMOTE or WOO_COMMERCE
70           dataProvider: 'LOCAL',
```

Set GCM id

Note that in order to push notifications to the registered Android devices, you will need to set the `gcmId` (GCM project number) in `Gruntfile.js` as shown below:

```
Gruntfile.js
59      },
60      development: {
61        constants: {
62          ENV: {
63            name: 'development',
64            youtubeKey: 'AIzaSyDael5MmCQa1',
65            ionicCloudApiKey: 'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJqdGkiOi
66            ionicSecurityProfile: 'pushdev', // https://apps.ionic.io/app/
67            ionicAppId: 'eca7b2b5', // '2113c758',
68            gcmId: '773185234778', // Google Developers Console
69            // LOCAL or REMOTE or WOO_COMMERCE
70            dataProvider: 'LOCAL',
```

Send a Push Notification

Please, bear in mind that you can send push notifications only to real devices. In order to do this, you can use [POSTMAN](#) or the UI of the Ionic Platform.

POSTMAN

To send a notification to all the registered devices, you should use the related [API endpoint](#) and make a POST call. Also, you should have the following headers. For the authorization use the API token you created previously.

The screenshot shows the POSTMAN interface with a POST request to `https://api.ionic.io/push/notifications`. The Headers tab is selected, showing two entries: `Content-Type: application/json` and `Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJc`. A red speech bubble points to the `Authorization` field with the text "API token".

Set the body and place your certificate name where highlighted:

POST https://api.ionic.io/push/notifications

Body (JSON application/json)

```
1 {  
2   "send_to_all": "true",  
3   "profile": "clouddev",  
4   "notification": {  
5     "message": "This is a test push"  
6   }  
7 }
```

You may need to troubleshoot in case of a failed push notification. In order to see the state of your push notification and any error messages use the related [API endpoint](#). The GET request should look similar to the following:

GET https://api.ionic.io/push/notifications/7c7b46a7-cb1f-4d04-88f3-d6742d851331/messages

Headers (2)

notification id

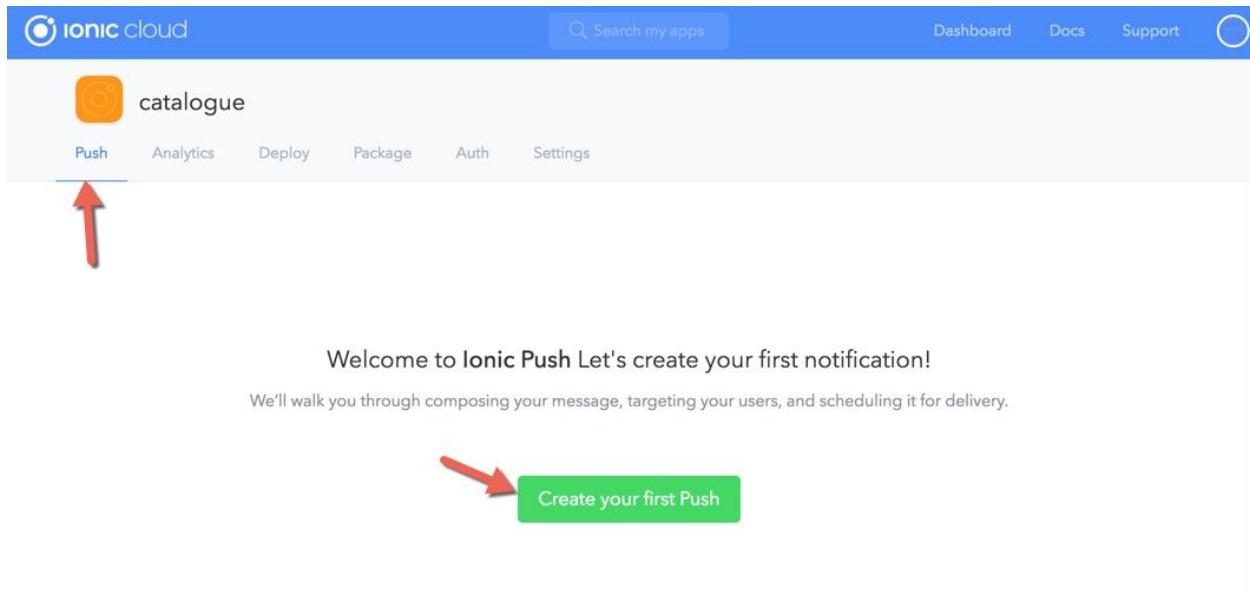
API token

Authorization: Bearer eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJqc

Content-Type: application/json

Ionic Platform UI

To send a notification using the UI of Ionic Platform, simply, create a new push notification.



Support

With regard to technical questions, new ideas and suggestions, you may use the dedicated form and choose the product your enquiry refers to:

<http://support.appseed.io/customer/portal/questions/new>

References / Links:

- [YouTube channel](#)
Periodically, video demonstrations and tutorials related to this product will be published in my YouTube channel.
- [Codecanyon User page](#)
You may contact us by using our user page on Codecanyon.
- [Titanium Templates Forum](#)
The Google Group that has been created for this product.
- [Quick Start Guide](#)
The online version of this document.

Thank you

Thank you again for purchasing my product. If you have any questions that are beyond of the scope of this help file, please feel free to email us to our [Support Centre](#).

--- *The Appseed team. appseed.io*