Software Architecture Documentation

JRC – MYGEOSS AirQ app for viewing AirSensEUR data

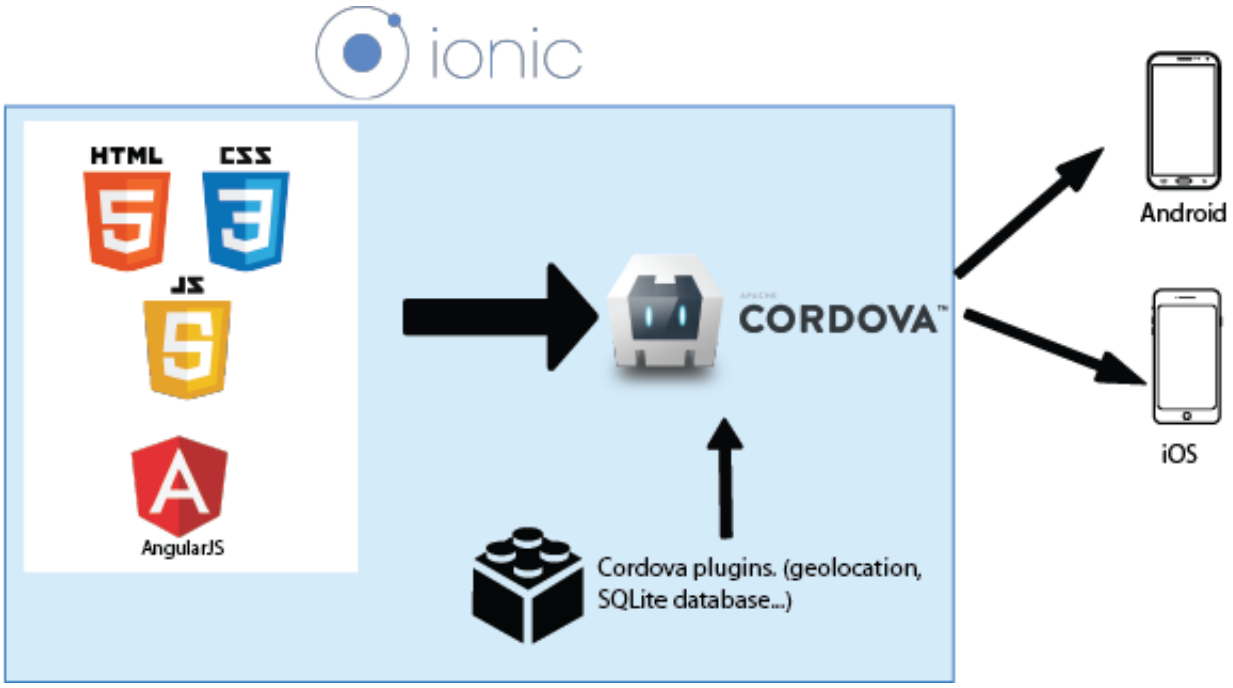
project 2016.3701

Table of Contents

1 Architecture View

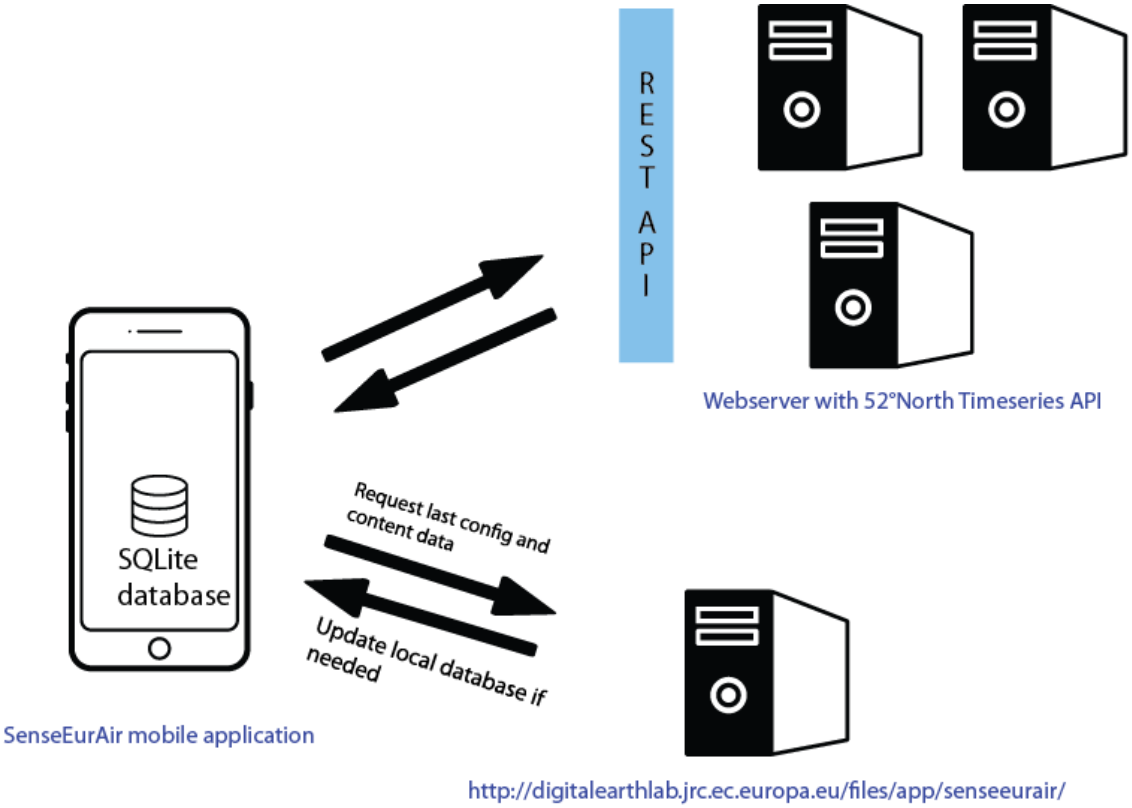
MYGEOSS AirQ (SenseEurAir) is a hybrid mobile application for Android (≥ 4.4) and iOS (≥ 8) that also runs on tablets.

This application connects to different external public API to retrieve data about air quality. This is also possible to change some of the content and configure the app from a server managed by the JRC.



Building hybrid mobile application with Ionic

Cordova and hybrid mobile applications use the native WebView of each platform to allow the use of standard web technologies. HTML5, CSS3, and AngularJS (JavaScript Framework) are therefore employed.



Different data sources

The application communicates with different REST API (including the JRC SERVER) to retrieve the air quality data. Documentation for the REST API can be found here:
http://sensorweb.demo.52north.org/sensorwebclient-webapp-stable/api-doc/index.html

Another JRC server is used to configure the application and update dynamically some content (like the about page and the Disclaimer…). The address is here:
http://digitalearthlab.jrc.ec.europa.eu/files/app/senseeurair/

2 Client local SQLite database

Persistent data are managed using a local SQLite database.

2.1 Table metaData

This table is to store persistent configs value like the last update date of the database.

Label	Type	Description
Id (primary key, unique)	integer	Unique identifier of the entry.
label	text	Key.
timestamp	integer	Value of the key.

2.2 Table airParameters

This is the data used by the application to display the information about some specifics parameters like the temperature or the CO2.

Label	Type	Description
Id (primary key, unique)	integer	Unique identifier of the entry.
idAirSenseur	integer	Unique identifier of the entry in the JRC AirSenseur network.
label	text	Label of the parameter.
formula	text	Formula of the parameter. (if exist)
unit	text	Unit of the parameter. (ppm for most of the pollutants)
source	text	Source text (pollutant only)
effects	text	Effects text (pollutant only)
limitValues	text	Limit values text (pollutant only)
icon	text	Value of the icon to be displayed.
description	text	Description text. (meteo only)
type	text	Pollutanst or meteo

2.3 generalContent

This table is used to store the static content of the app like the about page or all the modal text. (Acknowledgment, diclaimer..).

Label	Type	Description
Id (primary key, unique)	integer	Unique identifier of the entry.
label	text	Key of the entry.
description	integer	Content of the entry.

2.4 monitoringNetworks

This table contains all the data for the different monitoring networks with the mapping of the data between all of them.

Label	Type	Description
Id (primary key, unique)	integer	Unique identifier of the entry.
APIurl	text	URL of the REST API
label	text	Label of the source displayed in the app.
flag	text	Flag icon of the network displayed in the app.

selected	text	“true” or “false”. If the source is selected by the user in the app.
defaultSelected	text	“true”. The default selected networks sources, can’t be unselected. This is the JRC one
airParamsMapping	text	JSON string used to map all the parameters. More details in the programmer manual.

3 Application

3.1 Technologies Used

This application is built using the IonicFramework. This framework use Cordova to create hybrid mobile applications and comes with a list of plugins to use the native functionality of the device. The applications are implemented as a browser-based WebView within the native mobile platform allowing it to use the common web technologies, more specifically HTML 5, CSS 3, and JavaScript.

Ionic also uses the Angular framework and provide a number of custom directives.

3.1.1 Apache Cordova v.6.2.0

Apache Cordova is a library that is used to create native mobile applications using Web technologies. The application is created using HTML, CSS, and JavaScript and compiled for each specific platform using the native tools of the platform. Cordova provides a standard set of JavaScript APIs to access device features on all supported platforms.

https://cordova.apache.org/

3.1.2 Ionic v.2.1.12

Ionic is a complete open-source SDK for hybrid mobile app development. Built on top of AngularJS and Apache Cordova, Ionic provides tools and services for developing hybrid mobile apps using Web technologies such as CSS, HTML5, and Sass. Applications can be built with these Web technologies and then distributed through native app stores to be installed on devices through the use of Cordova.

http://ionicframework.com/

3.1.3 Cordova platform iOS v.4.2.1

https://cordova.apache.org/docs/en/latest/guide/platforms/ios/index.html

3.1.4 Cordova platform android v.5.2.2

https://cordova.apache.org/docs/en/latest/guide/platforms/android/index.html

3.1.5 AngularJS v.1.4.3

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML syntax to express your application components clearly and succinctly. Angular's data binding and dependency injection eliminates much of the code you would otherwise have to write, all from within the browser, making it an ideal partner with any server technology.

This is the core of the application.

https://angularjs.org/

3.2 Platform Requirements

iOS 8+, Android 4.4+, smartphones and tablets.

Access to features of the device:

- Access device storage
- Cellular network and WiFi
- Users geolocation

3.3 Deployment

The final deployment requires the publishing of the application on the Google app store and Apple app store. This is carried out by the Publications Office with information provided by the Author Service following the Publication Office document "*Guideline for mobile apps publishing on the European Union Accounts"*.

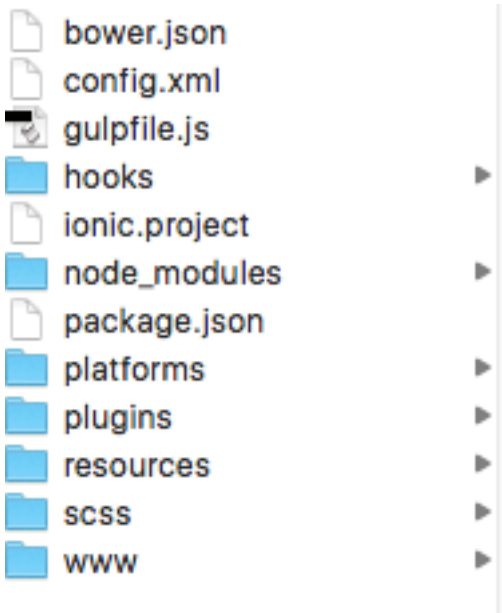
3.3.1 Folder Structure

We have two different structures. Cordova and Ionic provide one unique working directory (so both the iOS and Android versions use the same source code). The development code is inside the `www` folder. (More detail of this can be found in the *Programmers Manual*).

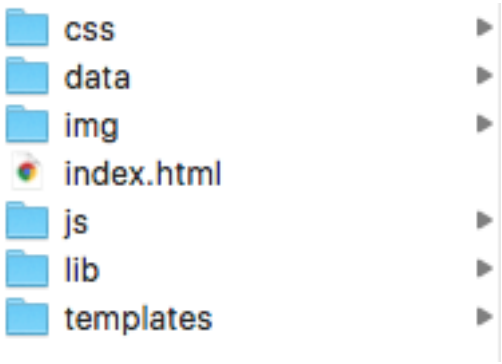
In the resources folder we find all of the icon and splash screen files, for the entire available platforms.

We use Sass to generate the CSS files.

The `config.xml` file contains all of the important information, such as the bundle ID and the version numbers of the applications. It contains the reference to the splash screen and icons for each platform, you can also find the platform version used and the list of plugin.

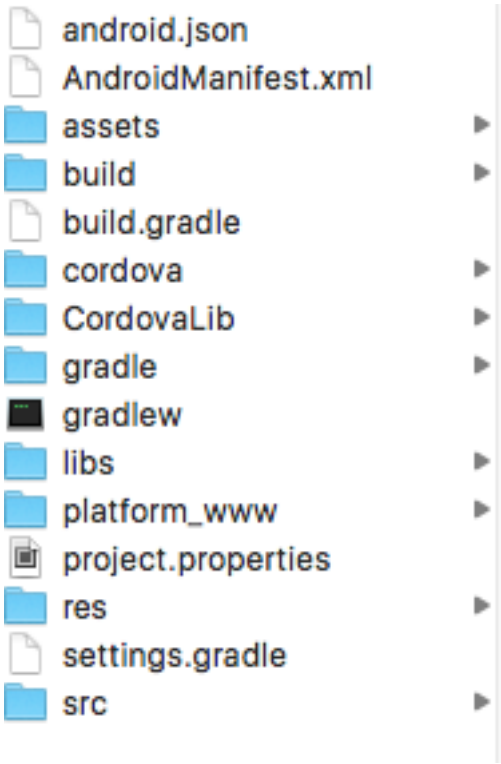


root directory

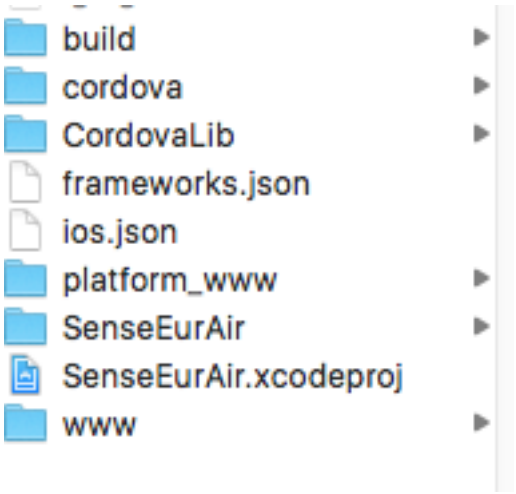


www directory

The code that is compiled to create hybrid app is inside the platform directory (Android and iOS). It is Cordova that generates these files, and they can be opened with Android Studio and Xcode, respectively. As a general rule we do not manage or change files in these directories. If a change is needed inside the platform folder, it will be mentioned inside the programmer manual.



Android platform



iOS platform

Further instructions on how to setup the developement environment can be found in the Programmer Manual.

END OF DOCUMENT

EPUB_Lot1_MAN_2016.3701_SoftArc_v010