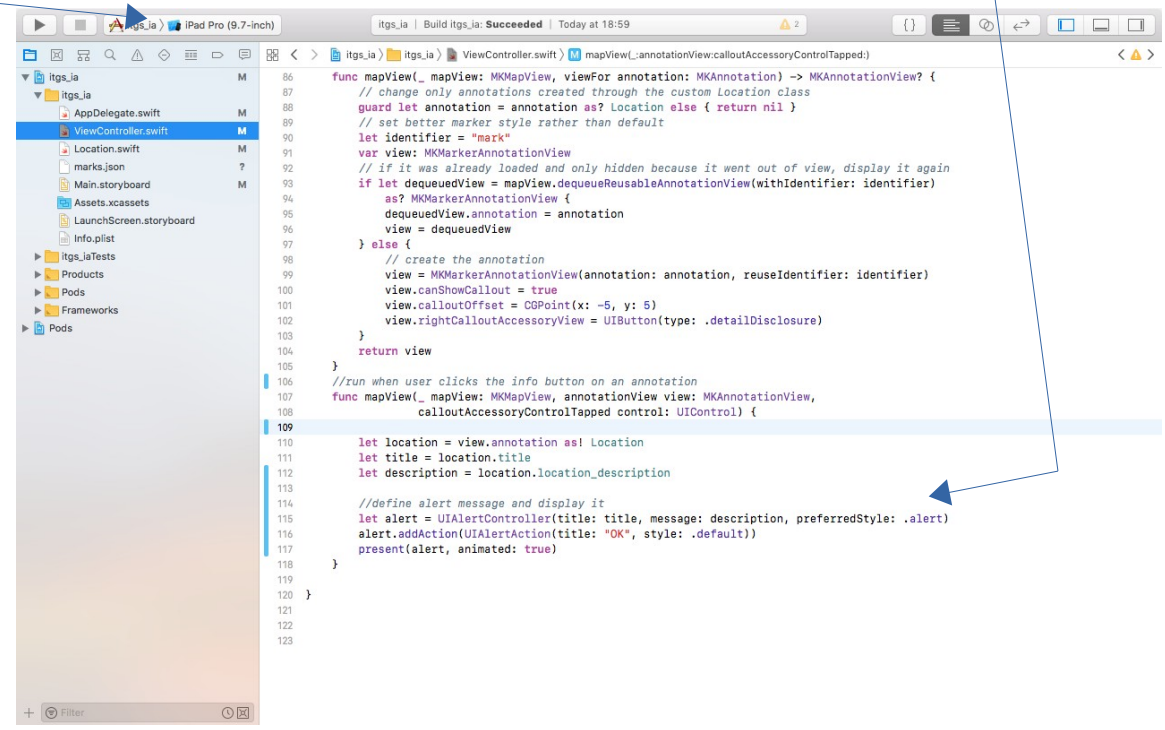
Criterion E: Product development

- Subroutines, parameter passing

Several subroutines were used throughout the code of the application in order to improve the structure, readability and optimization. To improve the user-friendliness of the application, each of the annotations on the map includes an info button in addition to the name and location. In order to make this button functional, I used the subroutine visible on the screenshot below.

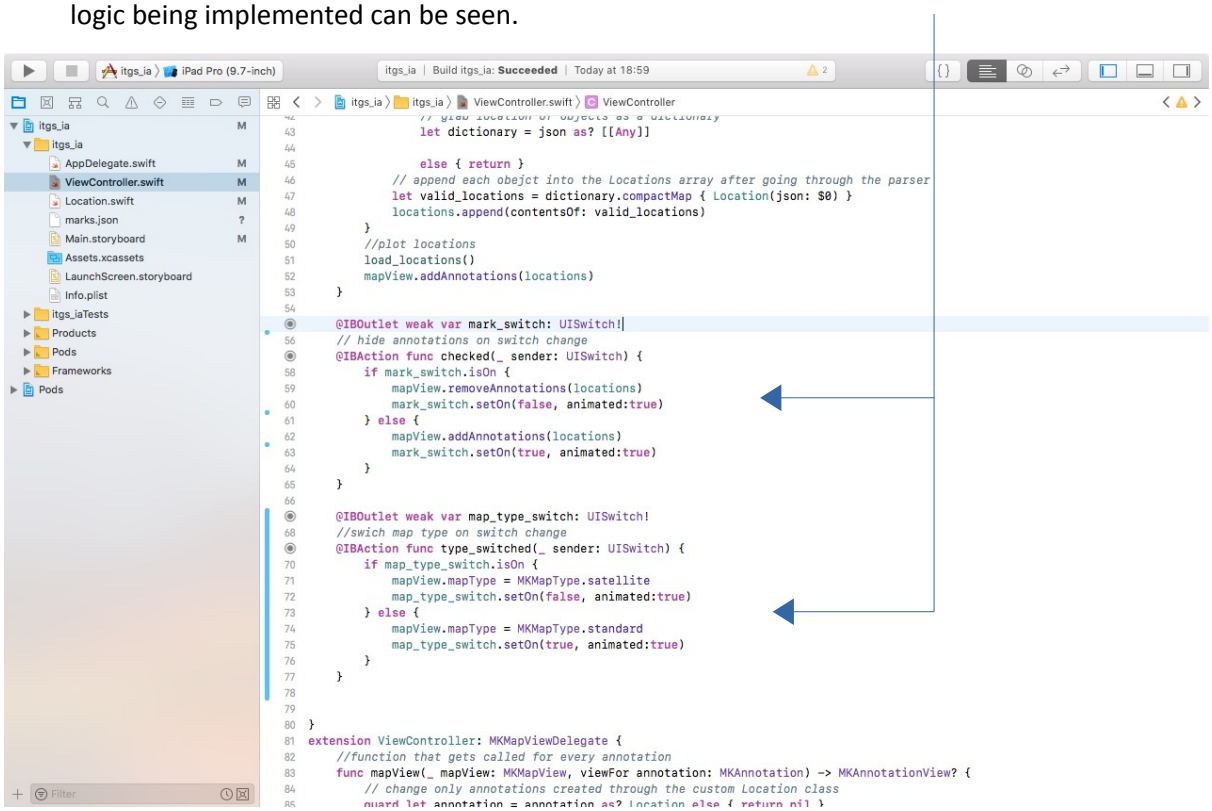
App being built and tested on the appropriate device



Whenever an info button is clicked, this subroutine is called and the object of the particular annotation gets passed into it as a parameter. The necessary information is extracted from the object, an alert is created and is displayed for the user to learn more about the location

- Loops, if–then, exit conditions

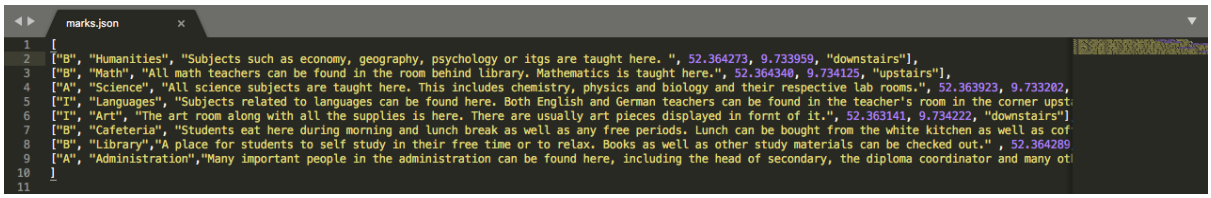
If-then conditions were used many times in the application as the building block of conditional logic. Exit conditions as well as loops were used mainly in the process of loading and processing the location data outlined in the section below. In the screenshot below, two examples of conditional logic being implemented can be seen.



While relatively simplistic in its essence, due to the complexity of the application environment even a simple on and off switch came with its challenges. In order to maintain the minimalistic and user-friendly design I set out in criterion C, I chose switch buttons rather than regular UI Buttons.

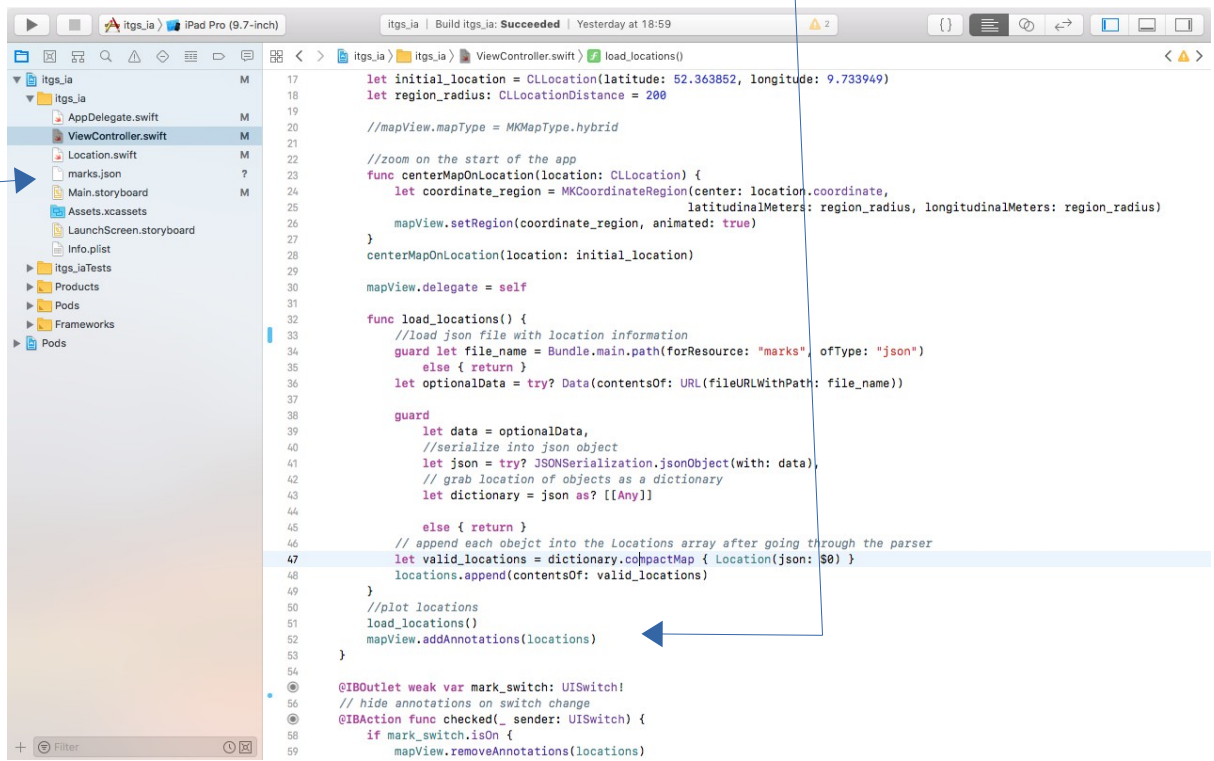
- File handling / Object definitions / Arrays

This is was a critical part of the application that I was required to implement for several reasons. Since the layout of the different part of the school might change, I was aiming for a simple way my client could change the information in it without having to interact with the code at all. This was done by storing the information in javascript object notation (JSON), a standard for such an application. Since all information is loaded from the file, it can be easily updated as seen on the screenshot below.



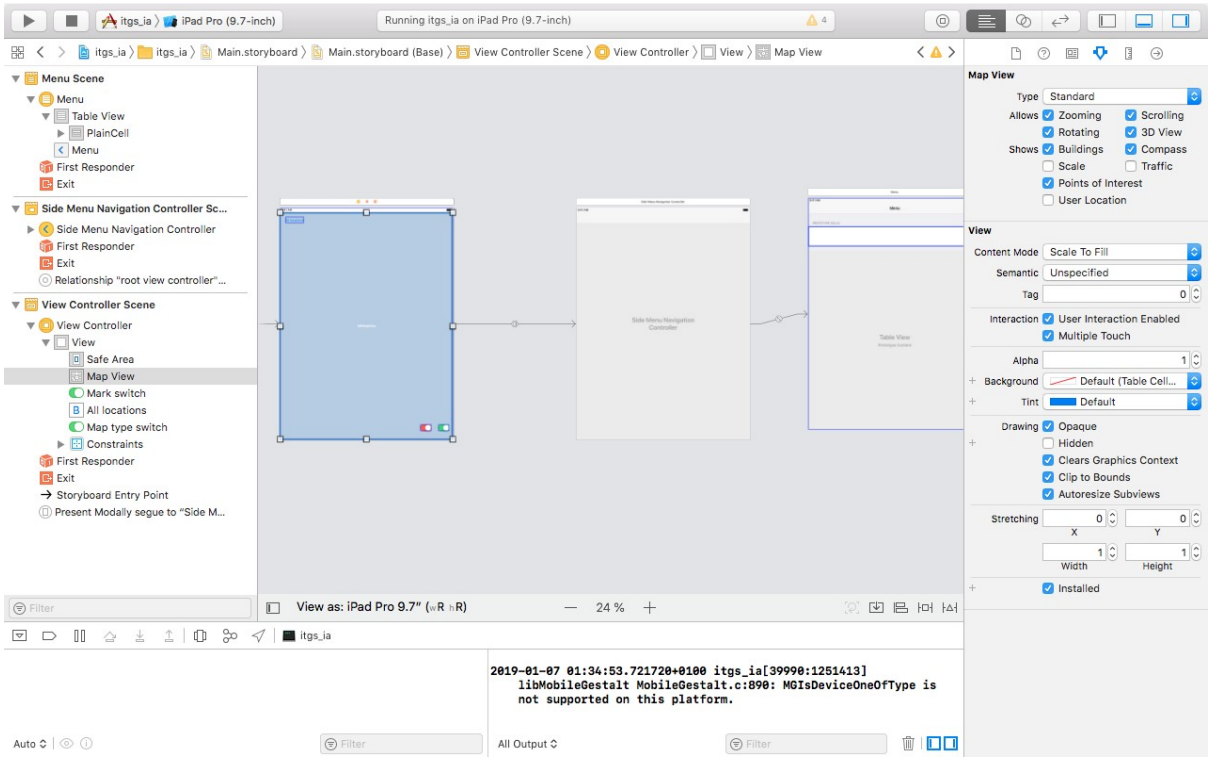
I decided to take advantage of the fact that Swift is an object-oriented language. This allowed me to represent each location as an object which can then be dynamically loaded and manipulated through the code in a native and efficient way. As shown on the screenshot below in the load_locations function, the JSON file is loaded, each object is then parsed through into a dictionary. It is then looped over and loaded into a swift array of objects which is later used to display the individual annotations on the map.

The project navigator showing the entire file structure including the marks.json file.



- Graphical user interface (GUI)

Naturally, an application cannot work without a graphical user interface through which a user can control it. In order to make it easy for the client and the app's users, I designed the GUI to adhere to the specifications set out in criterion D. This way, I would ensure that the client was getting what he wanted and I used the designs a developed earlier in the process. The interface through which a GUI is developed in Xcode is quite robust and offered me quite a lot of freedom in being as close as possible to the original design. The screenshot below shows some elements of this process.



Designing the GUI not only involved picking the particular elements and setting proper boundaries and layout setting but connecting each controller to the main view controller swift file were these elements can be manipulated through code. Additionally, the relationships between each controller had to be set in order for them to work together, like with the side menu, for example.

Word count: 583

Sources (MLA8)

Blewitt, Dr Alex. *Swift Essentials*. Packt Publishing Limited, 2016.

"Start Developing IOS Apps (Swift)." *Developer.apple.com*, Apple, 8 Dec. 2016, developer.apple.com/library/archive/referencelibrary/GettingStarted/DevelopiOSAppsSwift/.

Tam, Audrey. "MapKit Tutorial: Getting Started." *Raywenderlich.com*, Raywenderlich.com, 27 June 2017, www.raywenderlich.com/548-mapkit-tutorial-getting-started.