IOS Project 2018

Method

First a decision was made that this project would be about developing a sliding puzzle game. For a start a starting menu was created as our first view controller that included a start button, a stats button and an options button. When the starting menu was created people started to work on different things. This included an options menu that would handle music and sound volume inside the application. A stats board was also created for displaying how long time it took to solve a puzzle and how many moves that was required. A third thing that was being created was a "core data" database that is supposed to handle all the data used in the application, i.e. the difficulty of a puzzle, best completion time and moves made as well as the size and layout of the puzzle board.

After the options menu was created a class was implemented to handle sound and music. This class takes names of mp3 files and plays the sound/music. Sound effects plays one time and is used on buttons so the user gets a response when pressing buttons. Music starts when the application is started and loops while the application is active. The options menu has sliding bars that is used to adjust the volume of sound and music. To have an option to turn of sound and music mute buttons was created. These buttons mute the respective sound/music and disables the sliders until the mute button is pressed again.

Parallel to the sound the game menu was also being developed to make it so that we could show a puzzle on screen and have it listen and respond to user inputs which in this case was swiping in four different directions. At the same time the classes related to game logic were also being made, such as a board class and a rule engine class to check for completed puzzles and check if a puzzle is solvable.

To make it possible to use the pictures for a sliding puzzle game a picture have to be cut in pieces. A function that did this was found and implemented in the application enabling the use of any image in the puzzle.

Next up was a screen that allowed the user to select between a number of different images to puzzle with, created as a simple table cell view that registered the selected row and set the correct image in the game view controller. At the same time the UI was upgraded with proper constraints on all the different view controllers so that the buttons and labels worked on the different devices.

Furthermore, an API had to be implemented. Since this application is about solving puzzles of different pictures a decision was made that the API should get images from a website. So, through a HTTP GET request a bunch of image ids are obtained a JSON format and then saved in an array as integers. The ids then get used to retrieve the images from a url ending in ?image={id}. The 6 randomly chosen images then get saved in the database and made available for puzzling with. If the user does not like the retrieved images there is a button to replace them with six new randomly chosen ones.

Since the application should use some kind of hardware a decision was made to use the camera. This is made to make it possible for the user to take pictures that can be used for the sliding puzzle game.

The app also had the requirement of using some kind of push notifications. So, to enable this, a class that handles local notifications was made, initiated by taking in two strings, one for the title and one for the body. The created notification then plays when the app is in the foreground.

Who did what?

Database – Alfred
Image Functions - Alfred
Sounds – Simon
Game logic – Oliver & Alfred & Joacim
Camera – Simon
Notifications – Oliver
API – Simon & Oliver
Level generator – Alfred
Game difficulty – Alfred

UI:
Select Mode – Simon
Select Difficulty – Simon
Select stage - Joacim
Options – Alfred & Simon
Credits – Joacim
Game – Oliver & Simon
Win Screen – Everyone
Camera view – Simon
Constraints – Joacim & Simon

What couldn’t be delivered in time?

The goals of this project were from the beginning just being able to have a solvable puzzle on screen while also fulfilling the all the requirements for the different grades. The further the application was developed the broader the goals became, such as having different difficulties and board sizes. A stats screen was developed early in the developing process but later discarded since it was not needed in the application and there were other features with higher priority. Other than that, all features that was planned in the beginning of the project and during the project was successfully executed in the time frame.

If a deadline or the lack of owning MACs weren’t an issue, more time would have been spent on creating graphics to make the game more visually pleasant. Some options to add different borders or frames for the sliding puzzle would give the user customization options and make the application more attractive.

More sound effects are also a possibility. Maybe even made fully working on older devices and also having a landscape mode.

What went according to plan?

As mentioned before, the requirements were few and not very demanding. The only thing that was planned originally but never made it into the final product was a menu displaying Highscores/Statistics. The Method chapter in this report describes the working process during this project. Everything described there went according to plan even if it took some tries to get there.

What we discovered along the way.

This is the first group-focused assignment in DMP. This proved how easy and effective GitHub/Lab is for dividing smaller assignments and combining them into a working product. Also, during this project the involved developers had to learn the importance with working in different branches and what to do/ avoid to make a successful merge. Some of the developers also had to revisit/relearn some algorithms from previous courses, like calculating inversions or Manhattan distances. Another thing that was learnt from working on this project was how to efficiently deal with bugs or issues that prevented progress. If something appeared and was too time consuming for one person to fix, multiple people would group together and work it out. To make sure that everyone was always making progress and not getting stuck on an issue for too long.

Here is a visualization of the progress