Esko Naski, 0358522
Tatu Huttunen, 0358409
Joonas Kylmälä

Boiler Room development report

Table of Contents

1. Idea and Motivation

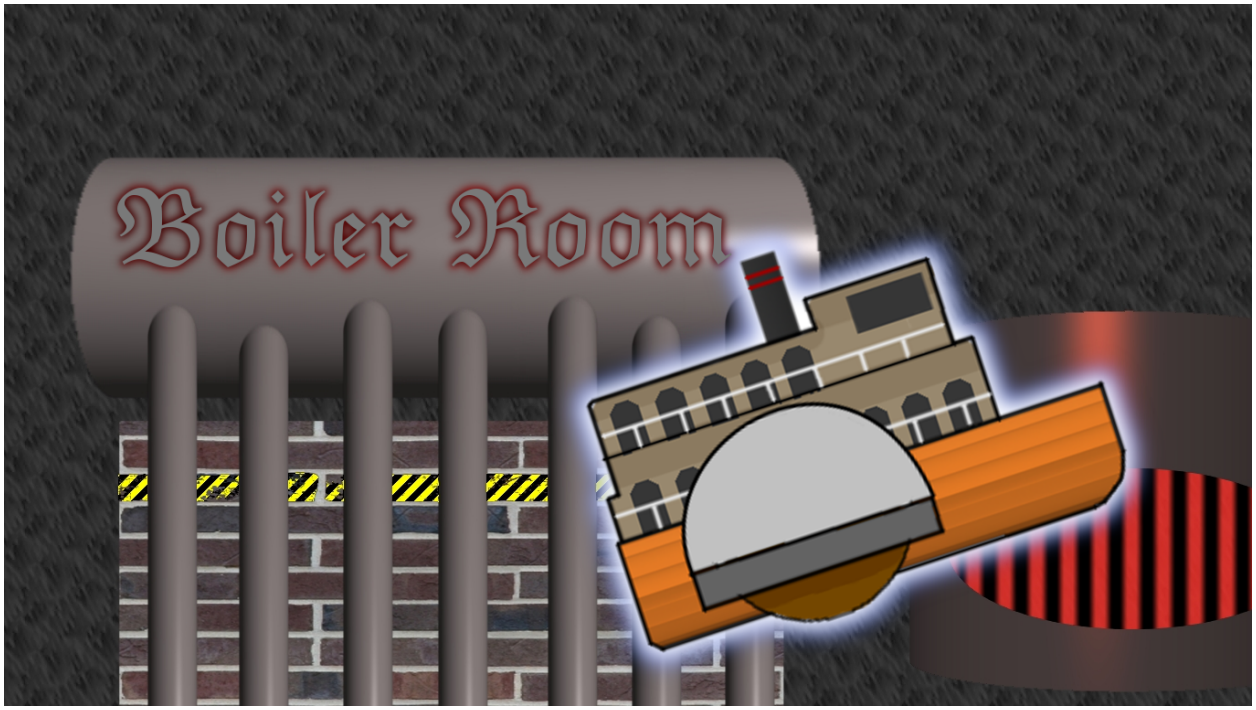
We wanted to do a simple project, given the limitations of the codecamp and our experience using Javascript and HTML. Tatu had previously worked on a couple of Javascript games while Esko and Joonas had next to none experience on Js. All team members had however some kind of programming background.

Our main goal for the codecamp was to learn about app developing for Windows 8. None of us expected the Boiler Room to be really nothing more than a learning project.

2. Arts

2.1 Graphics

The graphics for Boiler Room were made by Joonas Kylmälä and Esko Naski. Joonas created the main view of the boiler room, while Esko worked on the steamboat and the icons, splashscreen and such. Our main tool with the graphics was Gimp 2.8.



The idea for the main screen was to portrait an accurate view of a steam boats boiler room. The boiler room is really what the game is all about, so it was quite important to have it look nice. However as none of us had actually seen such boiler room, the result ended up a bit… intriguing. Also somewhat troubling was our team's complete lack of artistic talent and not enough photoshopping skills to compensate that.

In the upper part of the game screen there is a sideview of a steamboat crossing an ocean of some sort. The inspiration for the steamboat came from old images of such boats found on the Internet. Another moving particle on the screen is the quite simple looking pressure meter. As the pressure goes up and down, so does the indicator.

2.2 Sound

Implementing the sounds for the Boiler Room was an idea Tatu had at around 7 A.M. in the morning during the 24 hour codecamp. We all felt it would be something fun to do and that it would also keep us awake during the last long hours of the event.

The voices were recorded with Samsung Galaxy S3, as we couldn't find a proper microphone at that time. The soundtrack consists of three different clips: the melody was composed and performed by Tatu, as the white noise / windy sound and seagull screaming were done by Esko.

First we converted the .3ga-files to .wav-files on some free service provider on the Internet. After that we used Audacity to mix all the sounds to one beautiful song, lasting about half a minute.

3. Technical implementation

3.1 Tools

Game was made using javascript and HTML5 canvas. It is aimed for windows 8 tablets and PCs. Visual studio 2012 was used as IDE and visual studio´s own Team Foundation Server as version control. Games menu pages were designed using Blend.

We started working our project on top of one HTML5 game tutorial project provided in codecamp´s wiki page. tutorial can be found in following link: http://digitalerr0r.wordpress.com/2012/09/19/html5-game-development-for-windows-8-1-getting-started/ .
Tutorial project saved us time by providing a project which already had all ground work done and was ready to run out of the box. Tutorial project used CreateJS library to preload all assets and to store display data. CreateJS was unfamiliar to us but used functions seemed simple and effective so we ended up using them in our own game too. Used

CreateJS classes were stage, ticker and preload. Stage class is a container which renders its display list to canvas when called. Preload class was used to load assets to memory before starting the game. Ticker provides a tick to listeners at set intervals. it was used to monitor game loops framerate.

3.2 Code execution

When game is launched it first shows user the main menu. Currently main menu only has option to start a new game but we decided to implement it already to ease future development. Game also has menu views for game´s end states. These menus allow player to start new game or return to the main menu. In future settings and other functionality for example can be added to these views. Menu pages were made by Joonas.
When user hits play button, function prepareGame loads assets and sets up a new game. When everything is loaded startGame function is called and game loop start to run. startGame also is responsible for initializing counters which leak pressure and move the boat forward at set intervals. These intervals were calculated with javascript´s own setInterval method because unlike our createjs ticker it counts milliseconds not drawn frames.

Main game loop is very simple and its only job is to call functions to update game state and after that to call function that draws updated frame. Update functions firstly clears the canvas. This is done to prevent moving images from ghosting. Then update function checks if any game ending conditions have been met. These conditions are pressure getting too high or low (checked with function checkPressure) and boat reaching its goal (checked with function checkBoatPosition). If no game ending conditions have been met draw function is called and new frame shows up on canvas. Drawing new frame to canvas can be done only by calling stage´s update method because we used createjs´s stage container. If player wins or loses, function gameDirect calls corresponding menu page to be shown.

This game is simple and so is the code used to make it. All information needed to monitor games state can be stored in simple integer variables. Hardest part was to make the game to work on all different windows 8 devices. This meant different display resolutions and touch input. Touch input did not cause a lot of problems because microsoft provides an event(MSPointerUp) which is raised on mouse clicks and on touches. Using this event allowed us to handle input with ease. Different screen sizes were handled simply by scaling all assets by comparing devices resolution to our native resolution and then adjusting assets width and height by values obtained from the comparison. Also all adjustments to image positions were done using percentages of the screen size.

Conclusion

Calling Boiler room pretty simplistic is an understatement. It´s really simplistic no matter how you look at it. Gameplay consist of random mashing without any variations. That said we all are happy we managed to make it. Our first goal was to learn something from this project and we all did. Especially Joonas and Esko who had no mentionable previous experience in game development or javascript. Also considering the time we had to work on this project being able to make something that works and is showable was nice. Currently our game is not much to show for but by making some additional features and polishing some assets it might someday be an entertaining game.