Game Report - Puma Of Perth Road

Gavin Henderson - 150010848

May 12, 2018

1 How to Play

To play the the game visit https://pumaofperthroad.com.. There will be instructions displayed to the user so they can get to grips with the game but some of it is up to the user to discover. The games average playtime is 3-4 minutes but it has huge replay factor as you want to try and beat other players at the game.

2 Building the Game

The game goes through an unconventional 'build' process. Native javascript does not have any 'include' functionality so you can't write your code in different files. Obviously for a game of this size it would not be feasible to write the entire game so I use 'browserify' and 'watchify'. Browserify allows me to write my code with include statements and then it takes them and combines them into one file. Watchify allows this process to happen as soon as any file is changed. However, the game is hosted at "https://pumaofperthroad.com" so you don't have to go through this process to play the game.

3 Deviations from Plan

Throughout the development of my game I deviated from my plan in many ways. I realised that the they game I had originally designed didn't have enough features to actually play well. If I had only implemented the original features the game would have been slow but confusing. For this reason I added a few features as well as removed some.

One big change I made was the deviation from the story. My original plan was to play out the movie Wolf Of Wall Street however I moved away from that plot. I then decided to make it a more vague story of the come up of a stock broker. The story is still somewhat related so the naming of the game still makes sense.

One feature I added was the way that different parts of the game slowly reveal themselves to the user. This allows for a better progression of the game and allows for players to get used to one part of the game before having to deal with another.

4 Design Patterns

I used three design patterns I had originally planned on which were Dirty Flag, MVC and Loop.

4.1 Dirty Flag

> Avoid unnecessary work by deferring it until the result is needed.

The first is the 'dirty flag' pattern. I implemented it in a few of the 'view' classes. The purpose I used the design pattern is so that I didn't have to alter the view unless necessary. One example of it in use is in the PortfolioView class, which can be found 'static > view > portfolio. I didn't use it a textbook fashion but I still gained all the benifits from it.

```
// Only repopulate if stocks have changed
if(this.portfolioSize != this.portfolio.stocks.length){
     this.repopulate();
}
```

'this.portfolioSize' stores the size of the portfolio that the view has currently being displayed and 'this.portfolio.stocks.length' stores the actual size of the portfolio and only repopulates the view when needed.

4.2 Game Loop

> Decouple the progression of game time from user input and processor speed.

The second pattern is the Game Loop. In terms of abstracting the loop from the clock speed of the computer the browser actually does this for us. The browser provides 'setInterval' and 'setTimeout' functions which takes in a function and a time period given in milliseconds. This means that we don't need to calculate the time taken every tick of the loop. However, if we use 'setInterval' functions every time we want something to loop we will have no way to guarantee what order things run in and there is also unexpected behaviour. This is because it seems like you are creating a new thread but under the hood javascript just has one main thread with an event loop.

To solve those problems I abstracted the javascript funtctions into my own GameLoop. This can be seen in 'static > model > Loop.js'. My loop class provides an interface to make things run repeatedly. It has a separate loop for the view and for game items. The view loop runs 4 times every second, and calls the 'update' function of all the view items. This means the frame rate is esentially 4fps however the browser DOM doesn't work like that so its more like a maximum of 4 changes a second or data being at most 0.25s seconds old. I felt this refresh rate was acceptable for my game.

The loop for game object allows you to add functions to the loop and it lets you say how often you want the function to be run. It also lets you add one time functions and say how long after you want them to be invoked. The abstraction to my own class for this allows me to pause the entire loop with a boolean. This approach improves the game implementation as it means there is a clear place that has responsibility over the timing of the whole game and it is not up to each class to make sure that they sort their own timing. This means we are using a highly responsibility driven design which is desirable to increase extensibility of the game.

4.3 MVC

> Model–view–controller is commonly used for developing software that divides an application into three interconnected parts

I use MVC to make up the overall pattern of my entire game. Every component you see in the game is made up of three parts, a model, a view and a controller. This does slightly deviate as some aspects of the game are not seen so don't need to have a model and some aspects that have no login and are only seen so don't need a model. The main function of the program creates a controller of each aspect of the game giving them all access to the other controllers that they need. Each controller then creates a model and view for them and then acts as the communication between each part.

You can see this in action by looking at the 'static' folder. You can see the entry point of the game in 'index.js' that is inside the static folder. The folders are labelled with what they have in them so you can clearly see all the different components of the game.

5 'Game Aspects'

I used two separate game aspects but the more prominent one is 'Social Integration'. The other is some artificial intelligence.

5.1 Social Integration

The social integration in my game comes once you 'lose' the game. It allows you to login to facebook and then put your score on the leader-board. The Facebook login means that your score can be shown with your name and picture. The facebook integration also means that once you have logged in you can get a better score it will automatically add your new score in place of your old one. You can see the code for this in 'public > facebook.js'.

There was some issues setting this up as Facebook have recently made changes to their developer program to improve data privacy. It made it harder to register an 'app' to get access to their API. Luckily the API itself is incredibly simple once you have access so it wasn't to hard to add it to my game.

I feel this social aspect is incredibly important to my game as the playtime of the game is itself short. This means that I need to give users a reason to play again and again. The social feature achieves this by giving the player a goal so that they can beat their friends scores.

5.2 Artificial Intelligence

I also did a small bit of artificial inteligence however it is not my main game aspect. I created 'bots' that mean you can automate a manual process. The bots you can buy then buy/sell stocks on the stock market depending on how well they are doing. These 'bots' have an advantage over the player as they have access to extra information that is deliberately not displayed to the user.

If I had more time this is one part of my game that I would focus on improving. Currently their operation isn't overally 'intelligent' and the way that they are upgraded doesn't have the effect that I was hoping for. However, I am very happy with the implementation of the 'bot' feature as it is extremely extensible so lends its self to future improvement well.

You can see the code for this section of the game under 'static > bot-behaviour' aswell as the BotShop model.

6 Security

As my game is a web based game I took security very seriously. I host my server on a cloud computing platform which allows me to guarantee uptime. I also got a SSL/TLS certificate which allowed me to use a secure HTTPS server rather than an insecure HTTP server. I thought ensuring encryption was incredibly important as I was passing around person data taken from facebook. This task was quite hard to achieve as I hadn't done it before but I am glad that I added it as it adds to my game.

Security in terms of 'cheating' in my game is quite hard as the javascript runs on the players computer and can be edited on the fly as javascript is interpreted. This has pros and cons but in my case it meant that the player can edit all values including the cash values. However, there was still things I did to prevent this from happening. I used a build tool to obfuscate my code and 'minify' it so that it was totally unreadable. Technically, it is still possible to edit the values but it would be hugely time consuming. You also can't submit scores anonymously which means that if you do cheat you have to do it with your facebook profile which typically deters people from cheating if they cant do it anonymously.

7 Testing

7.1 Individual - Play Testing

Most of my testing consisted of my playing the game and checking for actual errors as well as making sure the game was balanced. I also made sure that it was possible to progress in the game every time you play it. I discovered lots of bugs in my game that were easily solvable through this method.

I feel like the fact that I played my game so much meant that I ended up with a very balanced game. I also feel like my game lends itself to being played over and over again and getting better each time you play it.

7.2 Group - Play Testing

As part of the play testing session I thought it would be useful to write a survey about the game for people to write brief thoughts about into after they had played the game. This allowed people to be honest in reviewing my game. The questions I asked were:

- Did you like the game?
- Did you understand the game?
- Was the UI clear to understand?
- What would you improve about the game?

I identified a few things during my playtesting.

8 Ethics

The ethics of my game mainly revolve around copyright issues involved in the graphics that I used. Specifically the windows theme is obviously protected under copyright. However, I have clearly modified the wallpaper so that it has a rainy sky rather than a clear one. This means that my use of the background and the theme is being used as parody which allows me to use the theme. The game also clearly doesn't provide a market replacement for windows XP which is another reason I can use the copyrighted material.

Ethics in terms of the age rating of the game it is appropriate for lower ages than initially planned. I planned for the game to have more explicit content like in the movie.

9 Things I would do differently

If was to do the game again I would slightly modify the technology stack that I used. I would still opt to use to create a web based game. However, I would chose a web based framework such as React, Angular or Vue.js allowing for easy binding of data to HTML DOM elements which is something I found myself spending a lot of time doing.

In the design stage of my game I would spend more time considering how the game actually plays as I think I didn't do this enough. My idea sounded good but as I implemented I found that the game did not make me want to play it. This is why so many features were added and removed from the original specification.

In the design stage again I would also maybe flesh out my classes more so that I knew how they interacted and stored references to each other. I struggled with the stage as I wasn't to sure about how javascript references worked but with this new knowledge my code-base would benifit from an overhaul. I would do things like make global references so I didn't have to pass objects through classes as this lead me to having huge constructors.

If I had more time I would be interested in experimenting with using real world stock data to influence the stock prices in the game. This approach may not work as the stock prices might not fluctuate enough but I would be interested into looking into it.