The Inversioning Documentation

# Introduction

This is a basic breakdown of The Inversioning project and how it worked out. Alongside some things I would have done differently if I had a chance to redo the project. This project was a solo project created by B00234203 so as such all work was done by me minus the audio and a lot of the images I sourced from online. I have a list of these in the copyright file on my bitbucket. The highscore table code is also a modified version from the labs.

I’ve added links below to my bitbcket repository, blog (you should already have these but just in case) and dropxbox hosting for my game. Unfortunately these have my name on them so leave them until last if you need to!

# Keyboard Controls

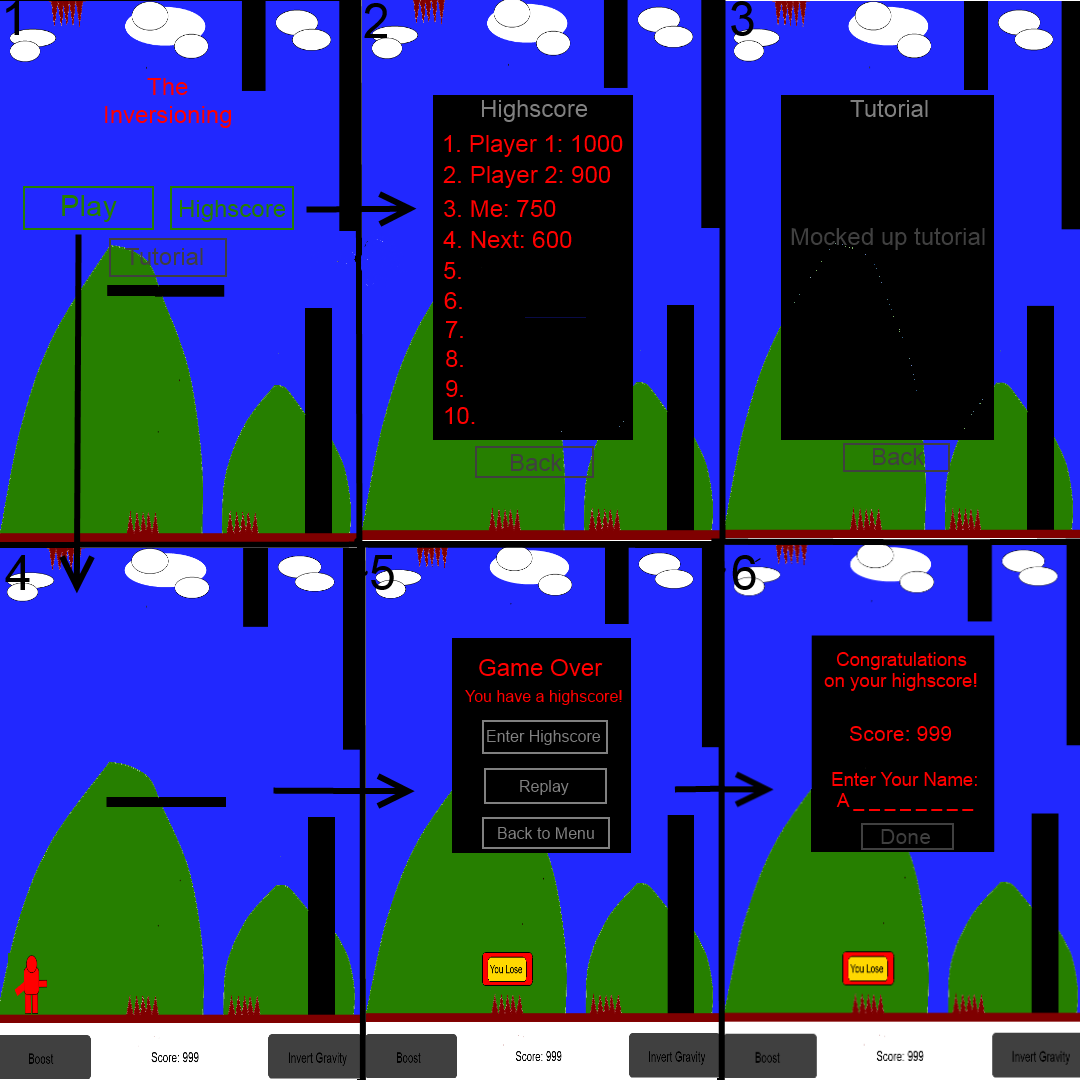
Whilst this is intended to be a mobile game and the on screen buttons are more than capable of being used knowing that you can use these two keys to play the game will be extremely useful if you’re testing it on a PC mobile emulator as it’s extremely difficult to play the game by mouse clicking the buttons.

|  |  |
| --- | --- |
| W | Invert Gravity |
| Space Bar | Speed Boost |

# Documentation vs. Reality

I believe that overall I’ve stuck reasonably close to the documentation schedule having finished all the tasks mentioned on it by the time they were specified except for the highscore table which I started later than I originally planned. Thankfully however it didn’t take anywhere near as long as I expected to do thanks to the great module example of a Google app server. This allowed me to get back on track. Despite completing the tasks by their expected date I decided to take a few extra days afterwards to improve the games randomization as the previous implementation was extremely lacking and made the game a whole lot less enjoyable than it now is. I then spent a fair bit of time cleaning up the games code as I wasn’t please with it which unfortunately pushed me over the hand-in date by a few days.

One thing that I mentioned was that I wouldn’t be using jQuery as I believed Phaser would be capable of handling an online web app without it. While this is true I unfortunately didn’t anticipate what I’d require to create the games online highscore table and as I found out during the creation of it, it needs jQuery to send requests to the server and receive the data. So as it turns out I ended up using jQuery in the end!



Using the picture from the Stage 1 hand in, I can say that that vast majority of the screen transitions work exactly the way as shown above there is in fact fairly little difference. The one outstanding change is that the bottom bar with the buttons on it doesn’t appear and move the screen up as we enter the play state instead there always remains a spot underneath the game area for the buttons to appear. The main reason behind this is that it would have required a lot more time to work out the values to push and scale each of the objects in the game to get the perfect window below the game area with the world scaling the same and it seemed like a minor feature in the grand scheme of things so it was pushed back in place of other more critical tasks. However I believe a middle ground would be relatively easy to implement within my current solution instead of scaling the world I could just change the y co-ordinate positions of all the floor items up and down to make space for it, however this wouldn’t be as aesthetically pleasing. Another minor changes is that the walls in the game are less sporadic in size and positioning, in fact they’re the same size and always get positioned the same distance apart it just randomizes if there is one or two of them in the same area and never changes their size. This is to mostly make randomizing everything a lot easier and reliable. However I believe scaling could be added in with reasonable ease by adding a random element to scale the walls on the Y! Other than that essentially everything else is the same or extremely similar in terms of screen transitioning and layout.

The biggest task in terms of getting this layout to work was getting the various world objects and players translating across the states, Phaser doesn’t actually allow its various “classes” to be passed through states it automatically deletes all of them on state switch. Phaser states also awkwardly don’t allow passing of variables between their own states (as far as I know and tested) so instead I had to make some “State” global variables which I’m personally not a big fan of at all. I alleviated this breach of OOP standard slightly by creating a GameWorldManager object which essentially acts as a holder for all of the worlds objects except the player (the player is just an state global array of variables such as position/rotation that we need to instantiate it in the same area in each state unfortunately), it essentially holds the positions of all the world objects and functions to randomize and recreate them as WorldObjects which is my wrapper class for the Phaser image class, this helps to encapsulate the majority of my code and keep the state globals down to a minimum. However I still feel this is a minor failure overall.

As for testing I used the Ripple emulator as mentioned in the documentation the game works on it scales and flips but has an audio issue that’s mentioned further below. I also got 3 classmates to test the game on their phones via dropbox it worked for 2 out of 3 of them. The one that didn’t work was a version of the iPhone.

Other than that I’ve been using tortoiseHg and bitbucket for version control and have been regularly pushing changes. I’ve also used jsDoc to create documentation for my project alongside the YUI compressor to minify the projects source code!

# Successes and Failures

One major change from what I originally planned and a failure that became a success is one that was recommended in the stage one feedback. It was to drop the idea of using Phaser physics in the game. And for the most part I have dropped it there are only two things that Phaser physics is used for in my game and that’s moving the world objects towards the player using flat velocity and for its collision boxes. The players’ gravity is just a flat value added onto the players Y position that gets inverted when the Invert Gravity button is pressed. I originally had the Phaser physics working to switch the players gravity up and down however I had to do a couple of odd work around’s to get it to work the way I needed it to and even after that the relatively accurate gravity wasn’t changing the players Y velocity fast enough to make the game in anyway enjoyable. So taking the recommendation into advisement and changing the way it worked was definitely for the best the game is now much more enjoyable.

My original randomization code for the world started out as a pretty large failure, the originally pushed code was enough for very basic randomization of two objects that get placed in two different planes but adding in walls caused a serious issue of things going into each other and the difficulty level was even more sporadic than it currently is. So in the end I went back and decided to fix it to be a lot more “procedural” (I believe that’s the word) than fully random. I done this by creating code that works as if the space between each set of walls is a “room” and a set number of obstacles or platforms can be generated within this space. It means the distance between WorldObjects can be worked out a lot more accurately so each one has its own “Area” to start in. It now has much less chance of objects going into each other so much so that it rarely ever happens. It’s also a lot more flexible so I could now add in more types of objects if I wanted to, all I’d have to do is edit the current randomization code slightly and increase the distance between the walls.

One of the biggest failures on the project was my original idea of getting the player to run through a 8000 pixel section and then reset-ing the player back to its original position and re-randomizing the area to give the effect of an infinite running space. I got it to partially work but it made almost everything a lot more complex than it was required to be. This finally hit home when I found out that trying to get a Phaser tile sprite working as a parallax background with the player running to the end of the world wasn’t going to work seamlessly and would leave very see-able spaces between the start and end of textures. To fix this I changed the way I thought of the world entirely to having the player stay still and the world move towards the player instead! This works pretty much perfectly making all the rest of the code extremely simple to work with.

Despite there being a number of failures that I turned into successes all of these problems and the whole project in general were made much simpler by the use of the Phaser framework so I believe that choosing to use it was a great idea and turned out to be very successful. And even though I didn’t plan any time to research it into the original documentation, I still spent a reasonable time beforehand practicing with the Phaser framework to make life easier. This turned out to be a great idea when it came to working on the project.

# Known Issues

* Flickering between state transitions – Unfortunately this issue is unavoidable using the Phaser state system, it’s due to having to re-instantiate everything and is extremely noticeable because the only thing that actually changes is the foreground!
* Double audio on Ripple emulator – I don’t think this is an issue that’s actually with my source code and more an issue with how the emulator works internally. As it works perfectly fine as a regular local web app and seems to work fine on the mobiles that play it! And on closer inspection the ripple emulator runs certain bits of code more than once even when it should only run once I.E an initialization method and I assume most mobile phones don’t do this.
* Occasional freeze on init – this is due to certain things not loading properly I believe, I was unfortunately unable to find the exact issue as its hard to consistently recreate the issue but it looks like some of the assets in the elements page don’t complete and end up flagging up an error.
* Slow load time when not run locally – I unfortunately believe this is due to the substantial size of the .ogg audio file I’m using it’s around 8mbs which seem to add 10 seconds or so onto the load time. This means that a blank Highscore table may display for a certain period of time, this is due to me instantiating it once during initialization so that each Phaser text object gets instantiated, it avoids an error that used to occasionally occur during highscore entry.

# Improvements

In terms of code improvements I’d start by removing the reliance on the Phaser state system I’d instead use my own similar to the Highscore system I currently have in place, this means that I’d be able to pass objects and variables around my own states without any problems. This would remove the reliance on state global variables. The only downside is that I’d have to handle my own Phaser object destruction for sprites etc. This in my opinion is a small price to pay for neater code and the possibility of avoiding issues further down the line if i wished to continue expanding it.

In terms of game play improvements I’d like to add some kind of flexibility to the difficulty or make it easier than it currently is. As much as I enjoy the difficulty level it’s at just now and the challenge it poses a lot of people won’t agree and get extremely disinterested pretty quickly. I think this could be partially attained by testing different numbers of platforms, obstacles and distance between walls in the game (or even better allowing the player to set them!), fewer obstacles would mean that the obstacles have more space to move in between the walls allowing more variety in where they spawn it would also mean that there’s less obstacles for the players to collide with and likewise increasing the space between walls would also help this. The extra platforms would just allow more areas to jump to, to make it easier to avoid obstacles etc.

Other than these two main changes I’d like to change the Menu, Tutorial and Highscore screen to look a lot more aesthetically pleasing as well as the death marker. These were mostly done this way as my artistic talents are pretty horrible I’d try to get an artist to help create replacements. I’d also change the size of the audio file or replace it with something else to decrease the load timing within my game. Or better yet at a pre-loading screen that shows a progress bar so that the user doesn’t believe it’s broken! This wouldn’t be too difficult as there is a Phaser example or two around to help out with it, the main reason I didn’t put it in was that I didn’t expect my app to take so long to pre-load up originally!

# Links

Bitbucket: <https://bitbucket.org/agozillon/the-inversioning>

Blog: <http://andrewgozillonblogmgd.blogspot.co.uk/>

Dropbox (takes a while to load): <https://dl.dropboxusercontent.com/u/87256252/The%20Inversioning/indexmin.html>