

Daniel Fowke

# Computer Game Development

Mathematical, Graphics, Platform and Delivery	5
Mathematical techniques, processes and Graphic processing	5
Platform and delivery	6
Visual Styles & Assets	7
Visual styles	7
Abstract	7
Voxel	8
Photorealism	8
Assets	8
Game Concept - Sol Defender	11
Story	11
Characters	12
Environment	12
Gameplay	12
Prototype Development	13
Interaction model	13
Narrative	14
Player Actions	14
Rules	14
Goals	14
Challenges	15
User testing and feedback	16
Game Design	18
Type of gameplay	18
Algorithm design	19
Visual styles	19
Arcade	19
Space	19
Minimal	20
2D	20
Bird's Eye View	20

Daniel Fowke

Highlighting, Selection Box	20
Assets, e.g. graphical, audio and video	21
Kenney's Asset Pack	21
Phaser Limitations	21
Gameplay features.	21
Select and Move	21
Legal	22
PhaserIO Legal	22
Kenney Assets Pack	22
Choosing	24
Choice of programming languages	24
Application program interface (APIs)	24
Intended platform/media for delivery	25
Planning and Timeline	26
3rd March	27
10th March	27
17th March	27
24th March - Test Day	27
Resources Needed	28
Hardware	28
Software	28
Test Plan	30
Constraints	35
Deadline,	35
Platform,	36
User.	36
Young	36
Educational	36
Communicating with the client	37
Developing the Game	38
3rd March	39
10th March	39
17th March	39

Daniel Fowke

24th March - Test Day	39
Feedback from initial testing	40
Movement / Controls	41
Identification by user	41
Design	41
Replayability	41
Improvements	41
Fun / Boring	41
Interface	41
Improving the Game	42
Future Improvements	44
Reviewing and Conclusion	46
Reviewing computer game	46
Quality characteristics	47
Skills, knowledge and behaviour	47
Skills	47
Knowledge	48
Behaviour	48

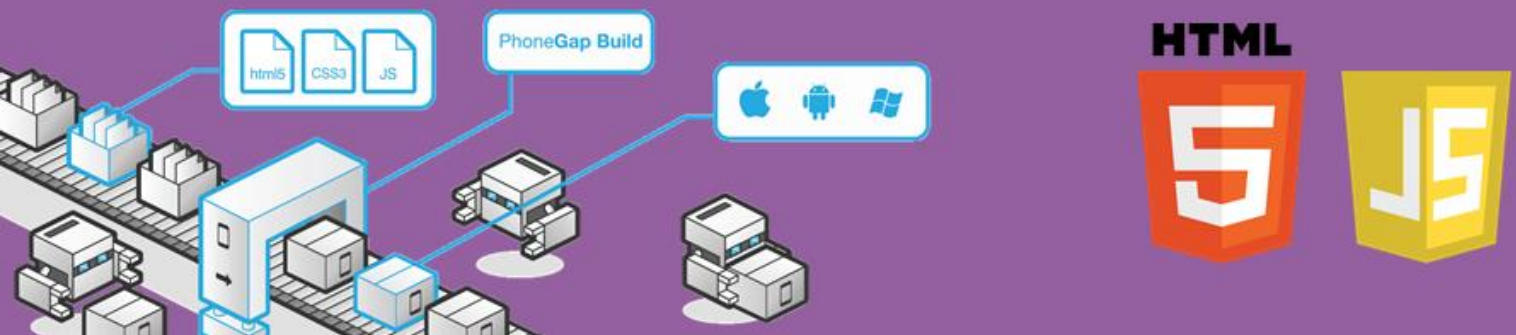
Daniel Fowke

# Computer Game Development

Daniel Fowke



# Mathematical, Graphics, Platform and Delivery



## Mathematical, Graphics, Platform and Delivery

### Mathematical techniques, processes and Graphic processing

**Mathematical techniques** and process can be done manually by myself where I could make a set of algorithms tailored to the game and how I want it to operate (**Realistic** or **arcade** as examples), though this method would be time consuming and 'on the move' research would need to be done for me to get the right level to make a set of mathematical functions that would control movement and even **physics** in my game. For **graphical processing** I would need to also make function from the ground up to allow it to display and run animation while also giving the control feature to optimize the game's **FPS** (Frames Per Second) again a very time consuming and a task I do not necessarily have the skills for.

The better option is to use an already existing game engine, this is based on the fact that my time-frame for making this game is very short. By using a **game engine**, all the general function I need to develop my game will be **already in place** and would only require me to configure what I need from the engine to use in the game. From this I

Daniel Fowke

could put **more time** into **picking my assets** for the game and **more on 'core mechanics'**

## Platform and delivery

Out of the gaming platforms my best choices would be Web or PC, this is because the simplicity that is needed to create a game for either platform, with these two platforms having the most open source software and resources for me to develop a game and simple inputs (Keyboard and mouse) to work on. Consoles would not be worth especially in my timeframe to develop with minimal online help to develop on those platforms and also including the expensive licensing fee

An advantage of having the game made for the Web platform would be that certain programming languages such as JS can be ported over to mobile for easy cross over. Allowing me to have touch on the game which is again a simple input as it can work as a mimicked mouse input.

Because of this my best option has been to use Phaser, an extension to **HTML5** and **JavaScript** for making **web based games**. With its own range of different physic engines to work off and a wide range of **modular libraries**, I can find just what I need and have **no 'excess'** to my game code, making it perfect for a **light and optimized game** as to pick a web game I need to make it easy and **small as possible for downloading** it each time the player goes to the website to play the game. For being able to port the app **from HTML and JS to a mobile app** can be done with **PhoneGap**, a service to help port existing HTML and JS with a few edits to be able to be a self-functioning phone app.

(Sources

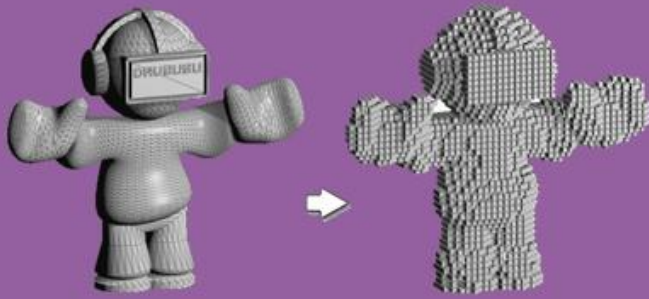
Phaser.io - <http://phaser.io/images/img.png>

HTML 5 - [https://www.w3.org/html/logo/downloads/HTML5\\_Logo\\_512.png](https://www.w3.org/html/logo/downloads/HTML5_Logo_512.png)

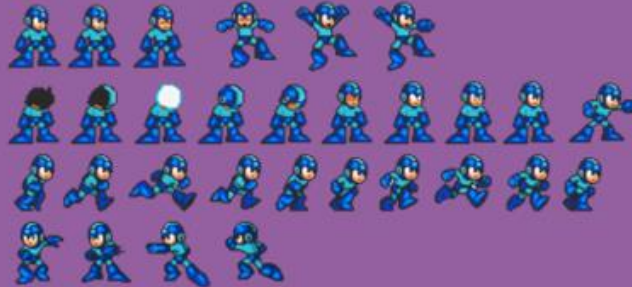
JS - <http://www.b2bweb.fr/wp-content/uploads/js-logo-badge-512.png>

PhoneGap - <https://build.phonegap.com/images/marketing/build-diagram.png> )

Daniel Fowke



## Visual Styles & Assets



## Visual Styles & Assets

### Visual styles

For Visual styles there is 3 styles;

### Abstract

Starting with the most simple being **Abstract**, this the most **simple** and **basic** of art styles that can be very common. Easy to make as it main would comprise of **geometry components** (Commonly squares) meaning rather than draw certain assets to detail they can be a mix of graphically displayed **shapes** from the engine and **pre-draw graphic** (Mostly **Pixel Based**) from the user than would be displayed with very basic designs, this would be the **least time consuming** and **very easy to implement** into the game - This will be my choice of Visual Style, using simple graphics and geometry made designs easy for your typical platformer or top down view game and also that simpler

Daniel Fowke

design styles work best on the **younger audience** as they **wouldn't have** the attention to detail for the full use of anything of **Photorealism** styles.

## Voxel

My other option is for something push the more physical environment of **3D** with **not too much** of demand to **time and dedication** compared to realistic or exaggerated graphics. **Voxel** is **block based** graphics where physical **3d objects** of fantasy and realistic elements can be made, just all **blocky** and **simple** especially for light and basic games. Though this may **not suit web based** as it would take considerable time to send the models over to the player through the internet each time, it could be viable if I intended for a **mobile** or **PC Game**.

## Photorealism

**Photorealism** is a style that would take extreme amount of **time, patience** and **attention to detail** to make. From make designs that **reflect** straight out of **real life** there would be a lot to consider, the processing of certain **textures** and **mapping** them out on **vector made objects**. This style requires too much precision as well as **Exaggerate**, the Realism style just '**hyped**' up a bit and **Cell-Shaded** which is a **cartoon design to realism**, making detailed and 3d objects have an **abstract** and **unrealistic** design,

## Assets

Assets is where I would design the entities and objects for the game, anything that is stationary or moves, to belonging to the player or the enemy I would have an asset pack to keep the collection, the purpose of this is so later in the game I don't have to do design as I go along but just add my assets into the game that I've done ahead of time to then have no worries back onto the design phase. My option for assets is very fortunate as I have gotten access to 'Kenney Game Assets' a collection of open to use game assets that have already been done for me though the drawback of this is that I don't have the control of very detail designs unless I go and do it myself it saves me time from starting from the ground up to make my game.

My option to start an asset pack starting from down up would allow me to have full creative control, making all the parts I need and finding my relevant reference to base my designs on would be perfect but like everything else that I would do from the ground up time would be lost, my main haul in the game design would be making a prototype and that would still be code heavy, my design of the game wouldn't need to be perfect as other games have shown as I would change it possible after more testing and feedback to get right into an asset making at the start could be very unnecessary.



Daniel Fowke

Using **pre-made** asset packs that have very strong relations to the game I would design could greatly **save** me **time** and **effort** in having deliver the **visuals** to the game but providing a **base** to work off to have the **game** and it's **functionality** to be able to be **done in less time**.

(Sources

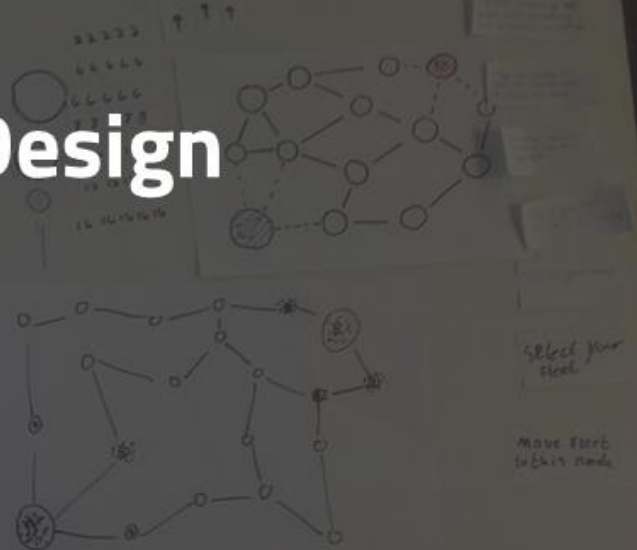
Toon Shader - <https://upload.wikimedia.org/wikipedia/commons/b/b7/Toon-shader.jpg>

Real -> Voxel - <http://media.gamersnexus.net/images/media/2012/features/voxels-vs-vertexes.png>

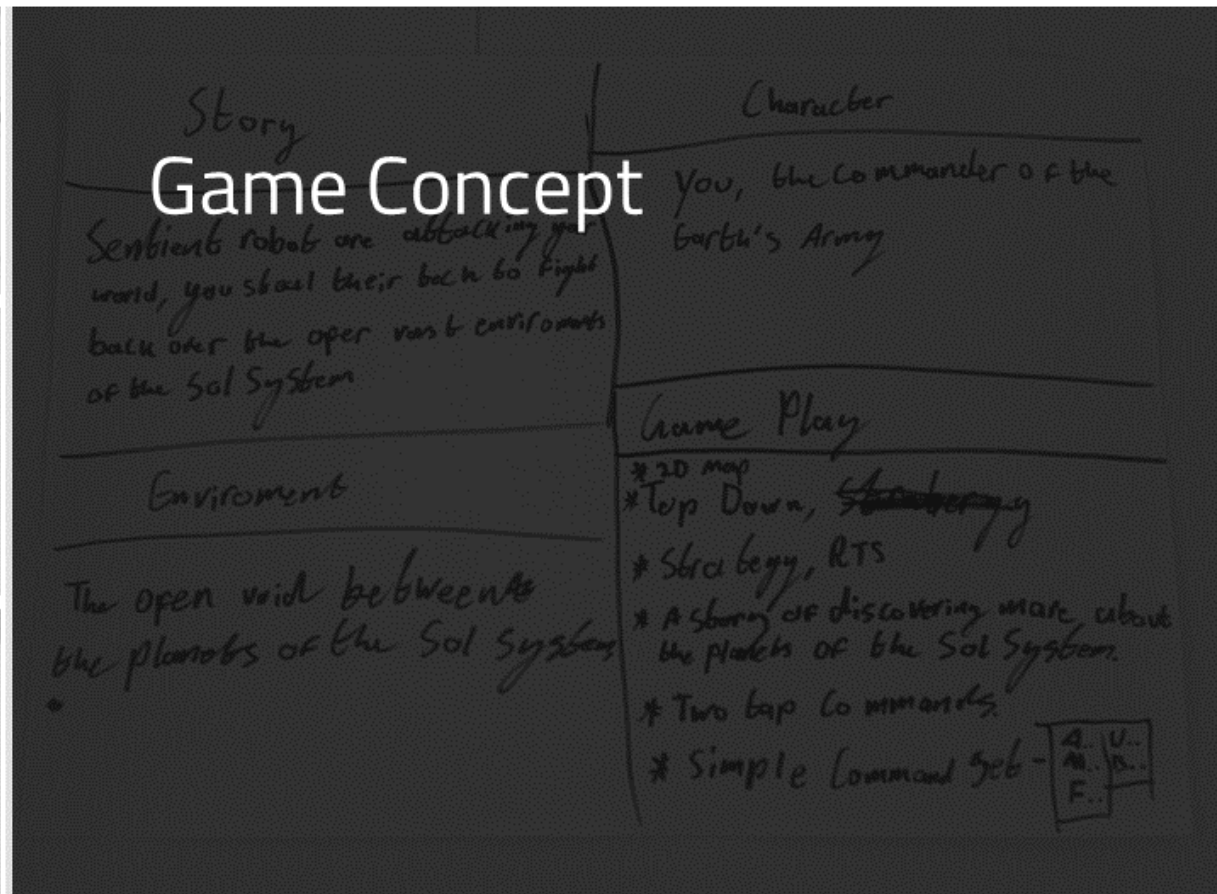
Spritesheet - <http://jslim.net/images/posts/2014-09-12-create-spritesheet-for-cocos2d-x-using-with-texturepacker/spritesheet-demo.png> )

Daniel Fowke

# Prototype Design



Daniel Fowke



## Game Concept - Sol Defender

To get started in the creation of my game I had developed my first game concept and made a very basic prototype to have had tested and feedback upon which has then helped shaped what I will be doing in the design stage and development of the game

### Story

- In the future ahead Year 2123 a big wave of robots have appeared in the Sol System with hostile aggression to the system, attacking Human Satellites and attempting to invade Earth
- The Robots have taken hold of all planets but Earth and as the Human reach defeat they launch their final hope
- The Pavilion the Human first and large Space Carrier that can sustain its own life and 'city', the player is the designated commander who is tasked with going around the Sol System and taking back their planets.

Daniel Fowke

## Characters

- You - The Commander of the Pavilion, you have control of its fleet and how to attack the enemy
- PAC - Pavilion Analysis Computer, the AI in the ship that help you learn about the Sol System, the planets you may travel to and also teaches you how to fight the enemy as they get tougher.
- Sentinel - The enemy 'supercomputer' the core of all the robots and the invasion to Earth

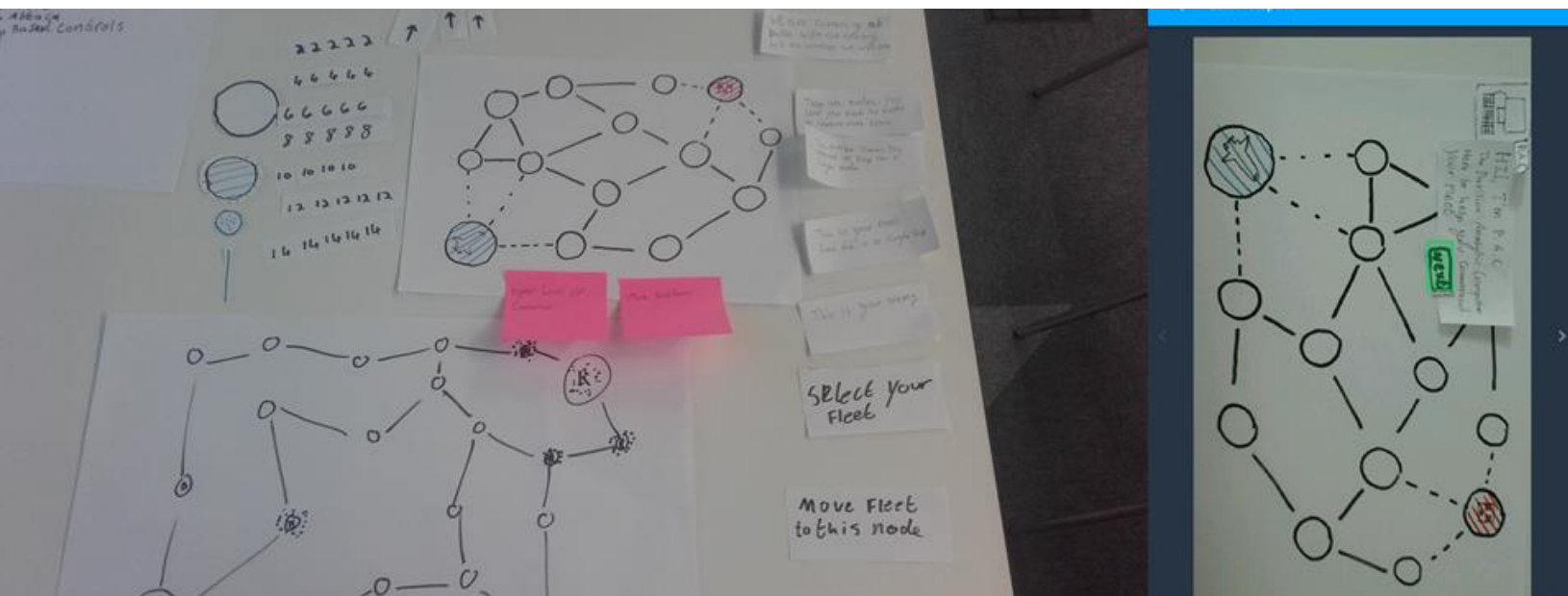
## Environment

- Sol System (Earth's Solar System)
- Asteroid field and vast open space
- Combat zone play zones (Space Debris e.g. broken ships, satellites)

## Gameplay

- Real Time Strategy - RTS
- 2D Bird's eye view of map
- Swarm based fighting (BOID)
- Basic Commands through clicks / touch (Move, Attack)
- Objective based
- Destroy enemy based win condition
- Single Player Game

Daniel Fowke



## Prototype Development

### Prototype Development

During the stage of writing up what I had for a working concept to making it an actual game made me think a bit more into the game and how it would work, with it all being in mind to be a game for a young audience, education in a sense and also more into the mechanics and control of the game. This was all then reflected into the paper prototype of the game, I developed a mini asset pack to supply the paper prototype and could work into the design stage as well.

#### Interaction model

Fleet / Squadron  
 Node (Control Points)  
 Movement commands

Daniel Fowke

## Narrative

To have voice dialogue in the game would be a bit time consuming and to produce even a viable voice over would require equipment that I do not have at hand, finally the fact that the prototype was on paper wasn't any more helpful to the idea of there being voice over dialogue in the game. Though I could carry the idea into the development phase. With that I was left to make dialogue boxes that displayed 'typed' text. This was me making an icon to represent the speaking character and then a box aside it with the text to show what said person is saying.

## Player Actions

The player should be able to select his/her fleet or squadron (undecided) and move it to nodes across the game space. This idea of simple select and move can work extremely well to reach a really simplistic gameplay feature in controlling the game.

When it came to making the player and his action be something that could be shown on a paper prototype I reach a wall, the problem was that my game even in a linear tutorial could have so many different states, that I wouldn't be able to create without the paper prototype process becoming unreasonable to do. As an RTS is all about different combination and that leads to there not being a 'best' way to represent the game without allowing the user total control of the game and its aspects.

As for the prototype development I decided the best idea to do would to make the tutorial of the game, the part of teaching the player how the game works and what they can do. This was the only point in my game where I could make linear control of the game in the active playing of it.

## Rules

Rules to consider was very little, a real time strategy game reward the most unique of thinker and coming to the idea of playing 'dirty'.

For the rules of the game, it was applied to how controlling nodes worked and the mechanics of capturing and defending nodes.

- You can only move to adjacent nodes to nodes that you own (Including start point)
- When capturing an enemy node, no enemy ships can be present near the node

## Goals

The objective for the player in each mission is to keep their base alive while they defeat the enemy by destroying their base

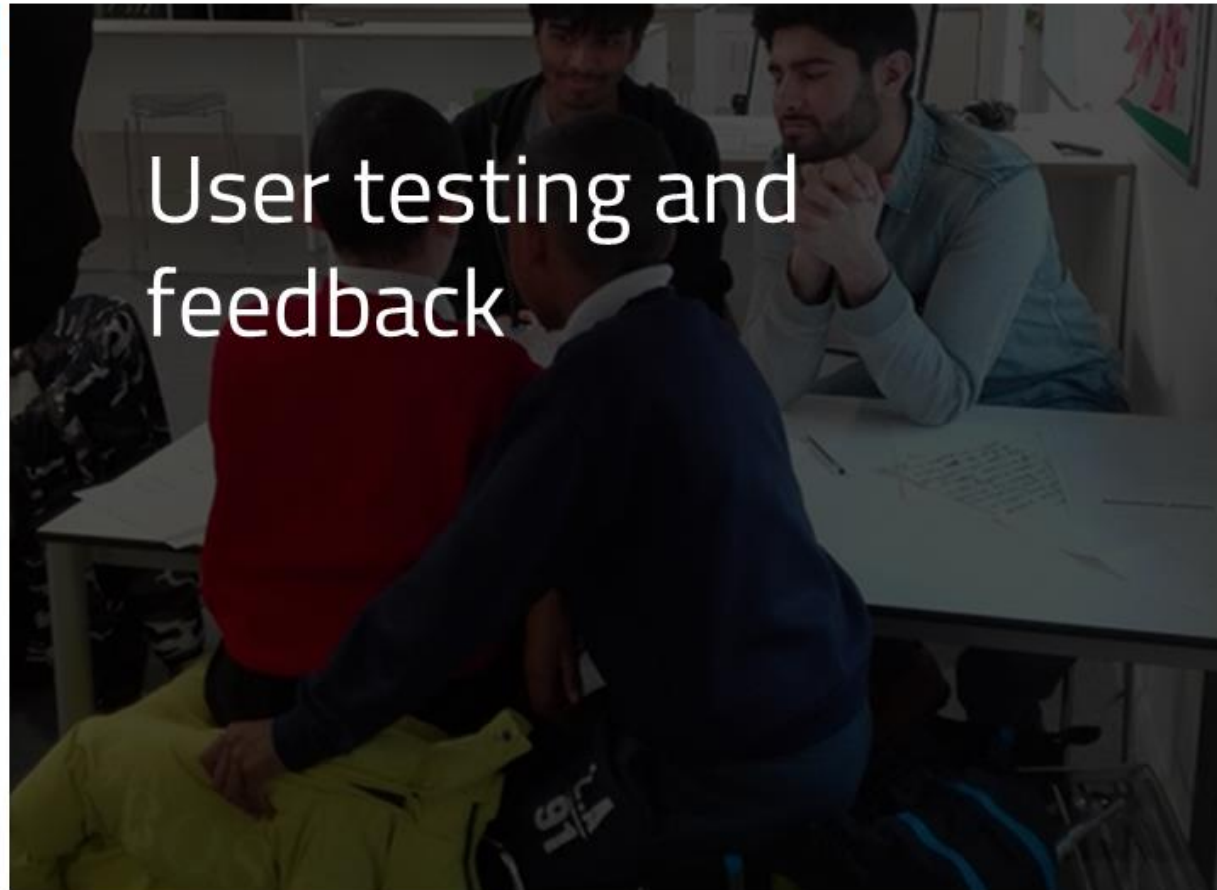
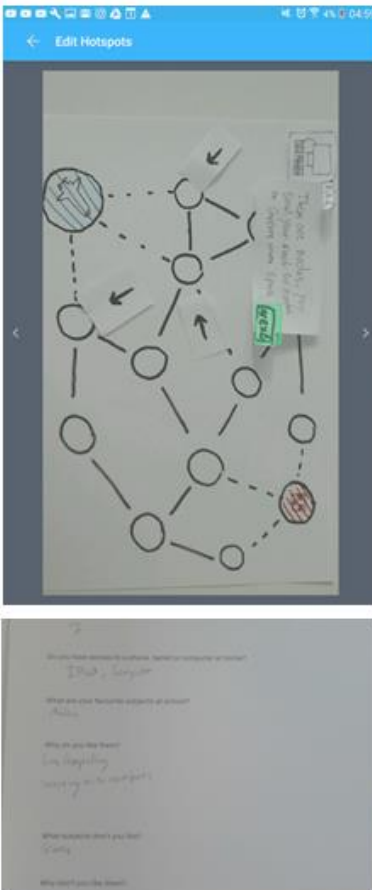
The goal of the game is very simple, to go to each planet and defeat the 'boss' and then to defeat the final boss

Daniel Fowke

## Challenges

As the player progresses further into the game the player will be handicapped by the enemy having boost in certain aspects (ships move faster for example) to give the player a challenge.

Daniel Fowke



## User testing and feedback

For the stage for my Paper Prototype to help setup the concept of the game and deliver it to other people was where I had to use POP by Marvel to create a paper prototype app that was used on my phone

I used an easy method to create the paper prototype by making a base layer that I could add assets that I could cut out and place on top to change the scene or update something without having to redraw the whole screen

For user testing I had 1 on 1s with a few Primary school year 4-6 students and from this received very positive responses, with asking them questions and having them run through the game they showed to be very interested in the idea.



Daniel Fowke

They are very interested with simplicity and from what I saw when they tested to prototype though they didn't read much or any of the dialogue they understood what they had to do then leading up to them understand how the game works, rather than reading what to do.

The response was the game needed something more but 3 of the Student's didn't really know what. Though on recommended to add a boost or some sort of bonus in the game to make it better. Leading to having ideas of modifiers in the game (Changing some elements in the game to give the player a boost or handicap, allows to expand into difficulty levels)

There are many points in relation to my game that I can list which was clear I needed to think about in the design of my game.

- Too much dialogue
- More 'creative'
- Not too far fetched
- Make the robots simple, (Not too complex in design (Simple robot with red eye as an example))
- Making the presentation of the game clearer

# Game Design

## Game Design

### Type of gameplay

Real time strategy - Putting your mind to the task to achieve victory is key to any game of this genre potentially requiring the most amount of thinking than any other game does the game require to outwit your opponent to defeat them

The game will be played from top down better known as Bird's Eye view and would mean the player see everything in the game at any given time. The game objective would be to capture all the nodes making the game a, 'domination' game mode when looking at the various types of games out there.

Daniel Fowke

## Algorithm design

Boid functionality, creation of swarm function means it would represent many entities without fully overlapping each other

From researching into Boid algorithms the best I found is a JavaScript made model in processing

Can be viewed here - <https://processing.org/examples/flocking.html>

From this code, I would need to make changes to the code to then apply to my game, there would be parts I would need to modify while some I would actually end up removing

The rest of my code would be for conditions and creating the nodes. Ideally the nodes would be colours to the side that has the most control on that node, or last touched it. So like a game of Tag with circular nodes

```
if enemyShips > playerShips then
  Node = red
else if playerShips > enemyShips then
  Node = blue
else node = neutral
```

## Visual styles

### Arcade

The game will not be in full detail, this is due to my lack of skill in designing and also the limited time I have to create the game, the best solution is to go to Arcade styles, this is something that I could quickly secure designs for either by myself or the use of online resources to get the general outline of the game functional

### Space

The setting of the game will be in the ever expanding universe, space. The main focus being the Solar system we all live in "Sol" my long term plan in designing the game would that there would be elements of all planets in the solar system, meaning I would have each planet drawn in a cartoon style. With that, also other assets to

Daniel Fowke

represent space would be, asteroids and satellites. These too could be represented in the same cartoony / arcade style

## Minimal

The design would have things to represent the setting but the best would be to reduce or make nothing but the core game assets stand out. This is to give the player the aid in being able to focus on the game without boring them out of lack of detail

## 2D

The game would be 2D as the design of this game would be easy and simple to implement for 2D as the idea is a top down view. Though assets design could be done in 3D this would just take a lot of time and I would lack the skill in this

## Bird's Eye View

Also known as Top Down view, the idea to give the player the feeling of control is done through this top down view, it's as if you as "elevated", giving the player the feeling of overseeing the game and in power of everything he can see (give it's not the enemy). This allows the player to understand that he leads in the game and can see what he controls

## NPC

In this game the NPCs are the ships, both the players and the Computer (enemy) as the player has no main character that is virtually represented in the gameplay. Instead he controls the NPCs that belong to his side by selecting them

## Highlighting, Selection Box

In order for the player to understand what he is doing, controlling and there will be responses to the player's actions. For selecting ships by clicking on them would the ships that he would then be controller would light up green, indicating to the player they have been selected.

Daniel Fowke

## Assets, e.g. graphical, audio and video

### Kenney's Asset Pack

For my project I have got access to Kenney's Asset Pack. A very large packs of various visual designs and audio clips that are made for PhaserIO, These have a range of assets from Platformers to RPG games with all the sprites and potential objects I would need for such games.

To this there is also an asset pack of a Space Shoot em' up game, however, these assets can fit straight into my RTS game since the Assets are really what I'm looking for with simple designed spaceships and asteroids.

### Phaser Limitations

Phaser isn't an AAA game engines, it is an engine made for very lightweight and simple games that can just run of the CPU mainly, no intense Graphic cards or APUs needed. So my limitations is how much of the assets I use at one time as well as the detail to be rendered.

## Gameplay features.

### Select and Move

The game is using a very simple click system, click to select then click to move, with this it forms the main core of the game's control function, maybe in the full implementation and changes would there be changes and more to just clicking

### Move along node lines

The way you navigate in the game has a simple rule to make the game playout better and form a better challenge, being that they can only send their NPC Ships along the pre-planned lines, if there is no connection from one point to another through these lines they wouldn't be able to travel across them.

Daniel Fowke

# Legal

## Legal

### PhaserIO Legal

PhaserIO is an open source game Framework, for them I do not need any legalities to use their product. I just download and use away. The most I can do is credit that I used their software by having it in the about section of my game once I get across to that part

### Kenney Assets Pack

Kenney Assets Pack is a community made asset pack under the license of "Creative Commons Zero, CC0" - <http://creativecommons.org/publicdomain/zero/1.0/>  
Their license state - "You may use these graphics in personal and commercial projects. Credit (Kenney or www.kenney.nl) would be nice but is not mandatory."

Daniel Fowke

For if I was to profit off making the game commercial would I also be present with possibly donating to support the creator of the asset pack out of courtesy

If I am to use any code examples online that do not belong to PhaserIO I would need to check on a case by case on the legality, in most case would they be open to use with the side of crediting their code out of courtesy and respect.

Daniel Fowke



# Choosing

Programming languages,  
application program interface (API)  
Platform



# PC

## Choosing

### Choice of programming languages

#### JavaScript

JavaScript is the language that can be used to make a Web Based browser game, this would also, given there is time to, be possible to port the game over to mobile as well, Either making the game responsive or recreating certain aspects to facilitate mobile device play

### Application program interface (APIs)

#### PhaserIO



Daniel Fowke

PhaserIO is a JavaScript framework that allows people to create games using game based functions and aids. This means that you can do quick import of assets each on a single line, and have functions such as rotate / angle that would do the whole turning and facing function all for you. This game engine would be usable for light games and mainly web based but possible for mobile devices

## Intended platform/media for delivery

### Mobile / Desktop

The simple point and click makes it perfect for both platforms, desktop being that you click the ships with the mouse / trackpad or mobile you would make presses based on the touchscreen input. The Main start will be having the game web based for desktop and the future plan would be to make it possible to have the game ported over to the mobile platform

(Source

JS - <http://www.b2bweb.fr/wp-content/uploads/js-logo-badge-512.png>

PhaserIO - <http://phaser.io/images/img.png>

PC - <http://i.imgur.com/7ZQgwGH.png> )

Daniel Fowke

# Planning and Timeline

## Planning and Timeline

Timeline, e.g. outlining which different assets are included and when different assets will be combined

From now till the first test day I have 4 week from the moment this section is being typed up. I have used that to label each of my lessons including this one to plan what I need to have ready

When I completed a stage I marked it with the JS file that I completed this with, made a copy and proceed to test and complete the tasks

Daniel Fowke

Date (Week Ending)	Objectives
3rd March	<ul style="list-style-type: none"><li>● Complete Timeline / Production Schedule</li><li>● Identify the assets to be used in the game</li><li>● Research BOID (Flocking Algorithm) and figure out what I need and don't need to apply it into my game</li></ul>
10th March	<ul style="list-style-type: none"><li>● Identify the limit to BOID count possible in PhaserIO and Web browsers without slowing the game the down (Logic to seeing this issue - the more individual BOID Objects the more calculations leading to more processing required per frame)</li><li>● Create BOID examples in Phaser</li><li>● Apply assets to BOID</li><li>● Create functions to make both sides attack each other</li><li>● Create death and respawn handling to the game</li><li>● Identify how to implement the methods of control from the user</li></ul>
17th March	<ul style="list-style-type: none"><li>● Create the first Level Map</li><li>● Complete the AI Development so the opposing side can move correctly following the same rules as the player</li><li>● Create random map generate</li></ul>
24th March - Test Day	<ul style="list-style-type: none"><li>● Create Interaction Highlighting for selecting ships</li></ul>

This planned out timeline looks very possible given the 4 weeks until the testing day and means I would be having a good 2 weeks being able to experiment and learn more about BOID while I develop, making it viable for the game.

Daniel Fowke



## Resources Needed

- Hardware
- Software

## Resources Needed

### Hardware

- Desktop PC / Laptop  
To create the game I will need IDE software that will allow me to create and edit JS and HTML code, be able to create my game using the PhaserIO Framework, also to test and play the game will I need to Desktop to provide this
- Mobile Device  
If I ever reach the stage where I have a very functional Minimal Viable Product on the Desktop Platform and decided to make a port for the mobile version would I need a mobile device for testing and checking that the port is being moved over the correctly

### Software

- HTML & JS IDE - Brackets

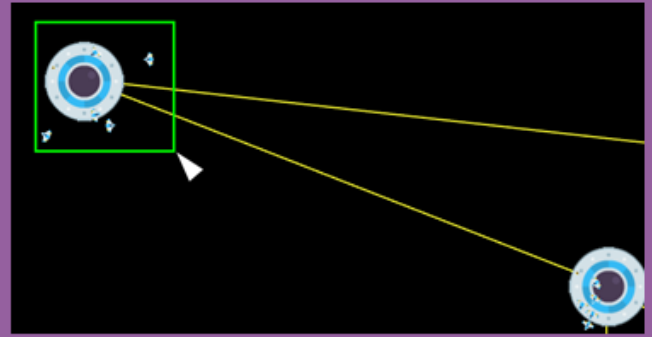
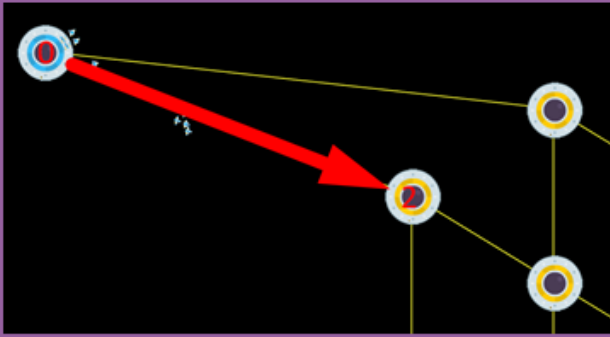
Daniel Fowke

Brackets is a web editor, being able to read HTML, JavaScript, CSS, JSON and many more languages related to web development. From this is also has an integrated preview feature that I can use to test multiple iterations of my game. This will be very useful in my development of my game.

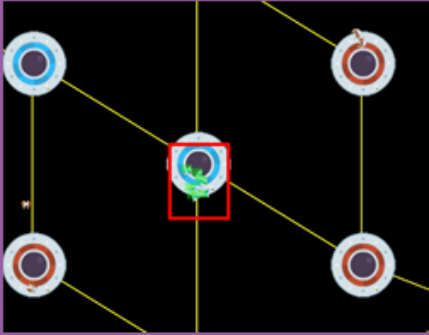
(Source

Brackets Logo - [https://upload.wikimedia.org/wikipedia/commons/thumb/4/4c/Brackets\\_Icon.svg/220px-Brackets\\_Icon.svg.png](https://upload.wikimedia.org/wikipedia/commons/thumb/4/4c/Brackets_Icon.svg/220px-Brackets_Icon.svg.png) )

Daniel Fowke



## Test Plan

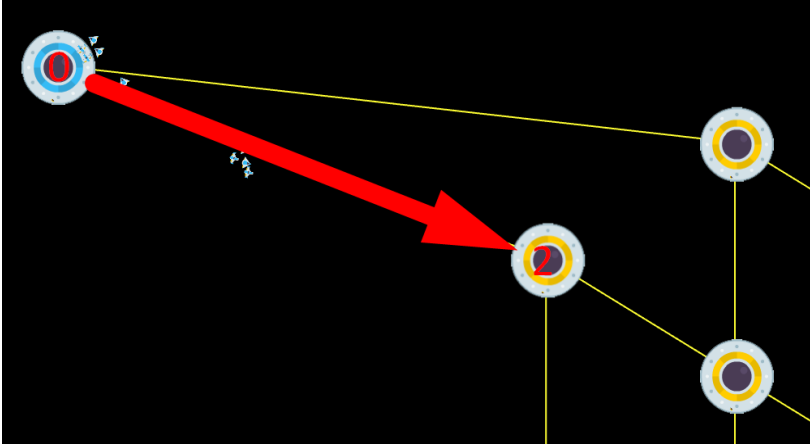


## Test Plan

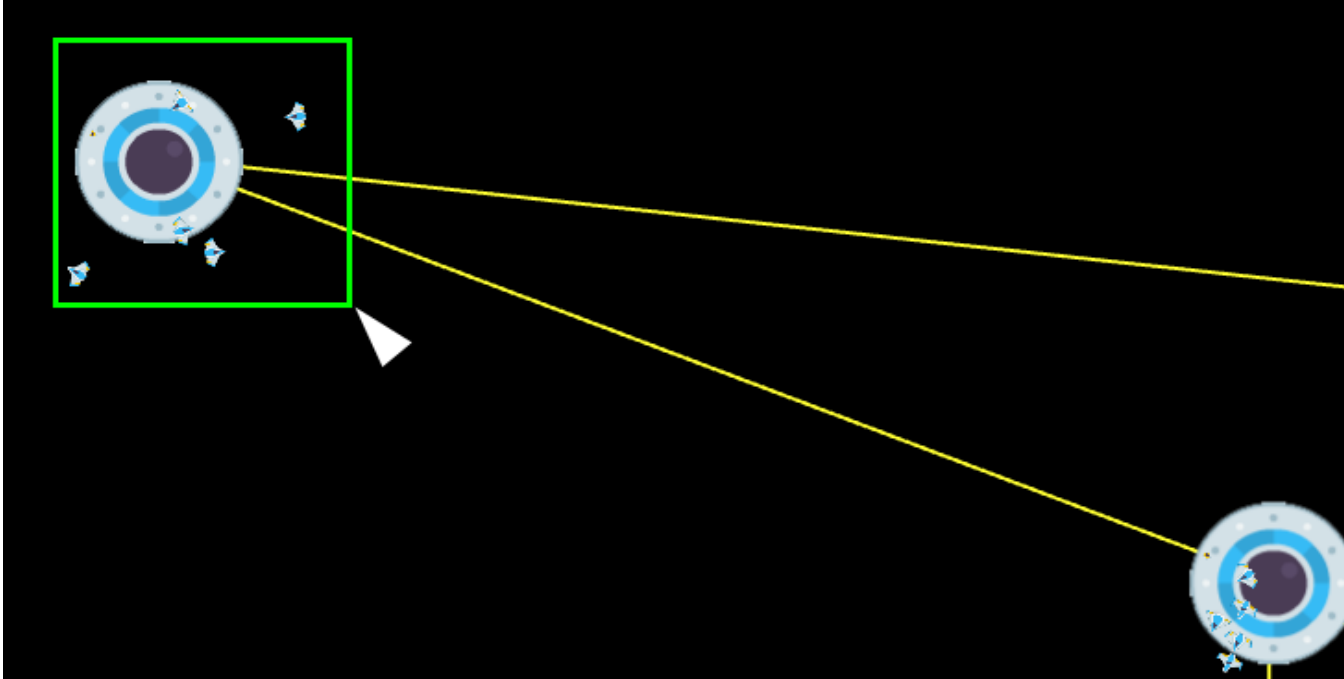
### Testing Plan and Testing Record

- Identifying how and what to test, e.g. producing a test plan, choosing test data
- Types of testing, e.g. effectiveness, functionality, performance
- Making improvements and/or refinements to solutions in response to testing

Daniel Fowke

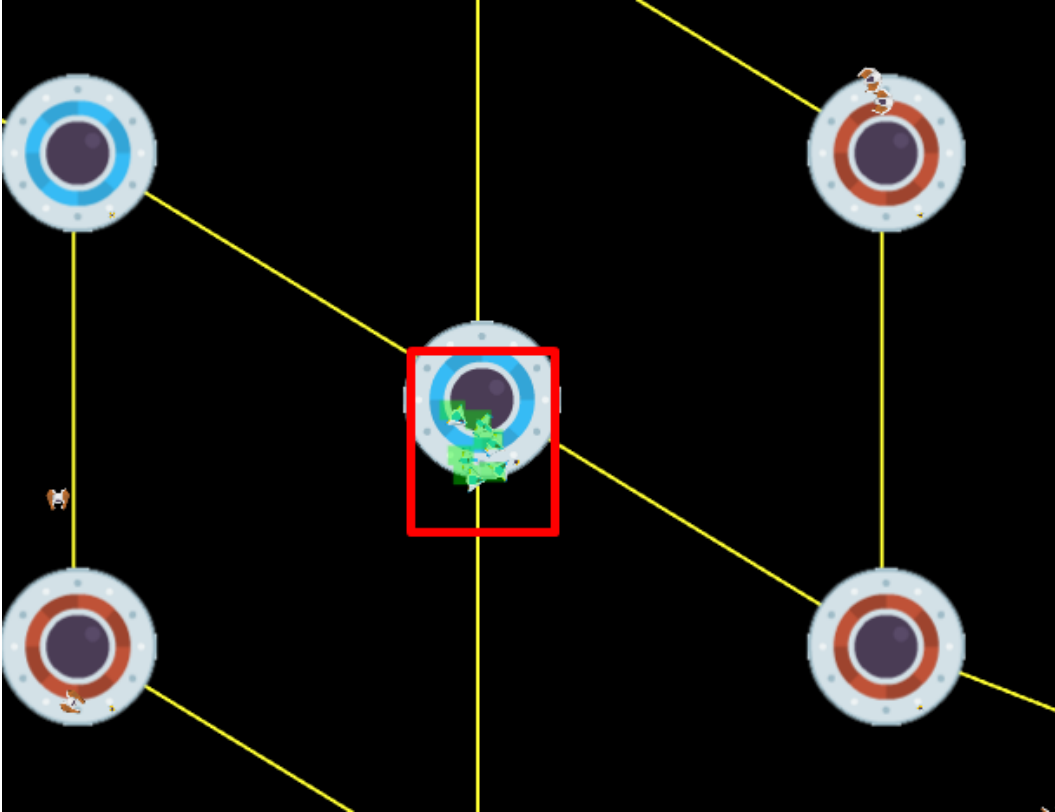
This part is done in the design phase before any coding is started				This part is done once the code is finished and assumed to be working. This could be done by the user		
<b>TEST PLAN</b>				<b>TESTING DOCUMENT</b>		
Test Num	Description	Test Data (valid, invalid, Erroneous Type)	Expected Outcome	Actual Outcome	Discussion	Code fix - improvement or refinement
1	Test that the selected ship will move to a node that it has a connection to	<i>Move ships at Node 0 (Start) to Node 1 (End)</i>	<i>The ships travel to the selected node</i>	<i>(Image Below) Ships moved to the mode accordingly</i>	<i>The ship are programmed to move to the correct node, I do not see any needed changes</i>	<i>n/a</i>
Outcome						

Daniel Fowke

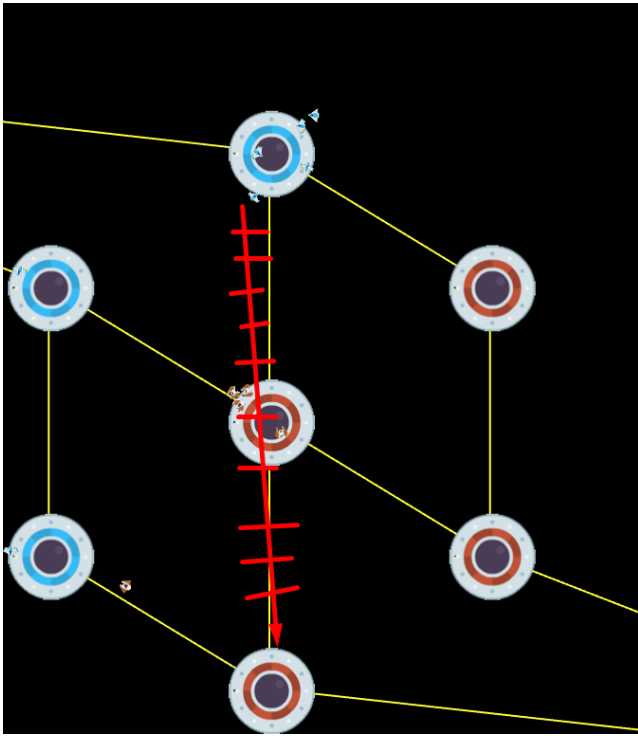
2	When dragging the mouse a green box would be drawn from the start and end point of the mouse	Drag a box covering node 0	A green box is drawn while the mouse is held down and disappears when mouse is released	<i>(Image Below)</i> <i>Green box was drawn until the mouse was released</i>	The drawing of the box is a little inconsistent and could do with some smoothing	n/a
Outcome						



Daniel Fowke

3	Selected ships would be coloured under a green box to indicate they are highlighted	Select ships at node 4 via dragging a box over them	The ships at node 4 would be highlighted green	<i>(Image Below)</i> <i>Ships were highlighted accordingly, green - meaning they will listen to any valid move commands</i>	Highlighting is visible but could do with some improving if given more time and able to understand debug controls	n/a
Outcome						

Daniel Fowke

4	Attempt an invalid move command	Move from node 1 to node 7 (A Invalid move order)	Ships do not follow the commands and remain at their position	<i>(Image Below)</i> <i>The ships do not move, the cross line indicating that was the move command and that it failed as this move order was invalid</i>	This is as it should be, I do not need to make changes for this	n/a
Outcome						

Daniel Fowke

# Constraints

## Constraints

### Deadline,

Timeframe - 4 Weeks

An average indie game would normally take longer than 4 weeks, as some major top hit indie games have taken about 2 - 3 years such as Undertale. The looks pretty simple but it takes incredibly long

My game's design needs to be simple and easy to expand upon to reach this deadline, I'll need the use of guides and following strictly down to what I can understand in the timeframe given, trying to use the more advanced functions in Phaser would take more time and be more subjectable for me making errors

Daniel Fowke

## Platform,

The plan for porting the game is: Desktop -> Port -> Mobile

To make the cross from Desktop to mobile I would need to assure that I do not surpass the power of a mobile device when making the port, it limits to the complexity of my game and since my game will be heavy on algorithms would mean it needs to be scaled to the correct power of the device.

To make these adjustments I would need to have time to analyse the limits and learn about both platforms during the game development phase, this would take a large amount of time and would mean I would constrain myself to making the game on Desktop first to meet time

## User.

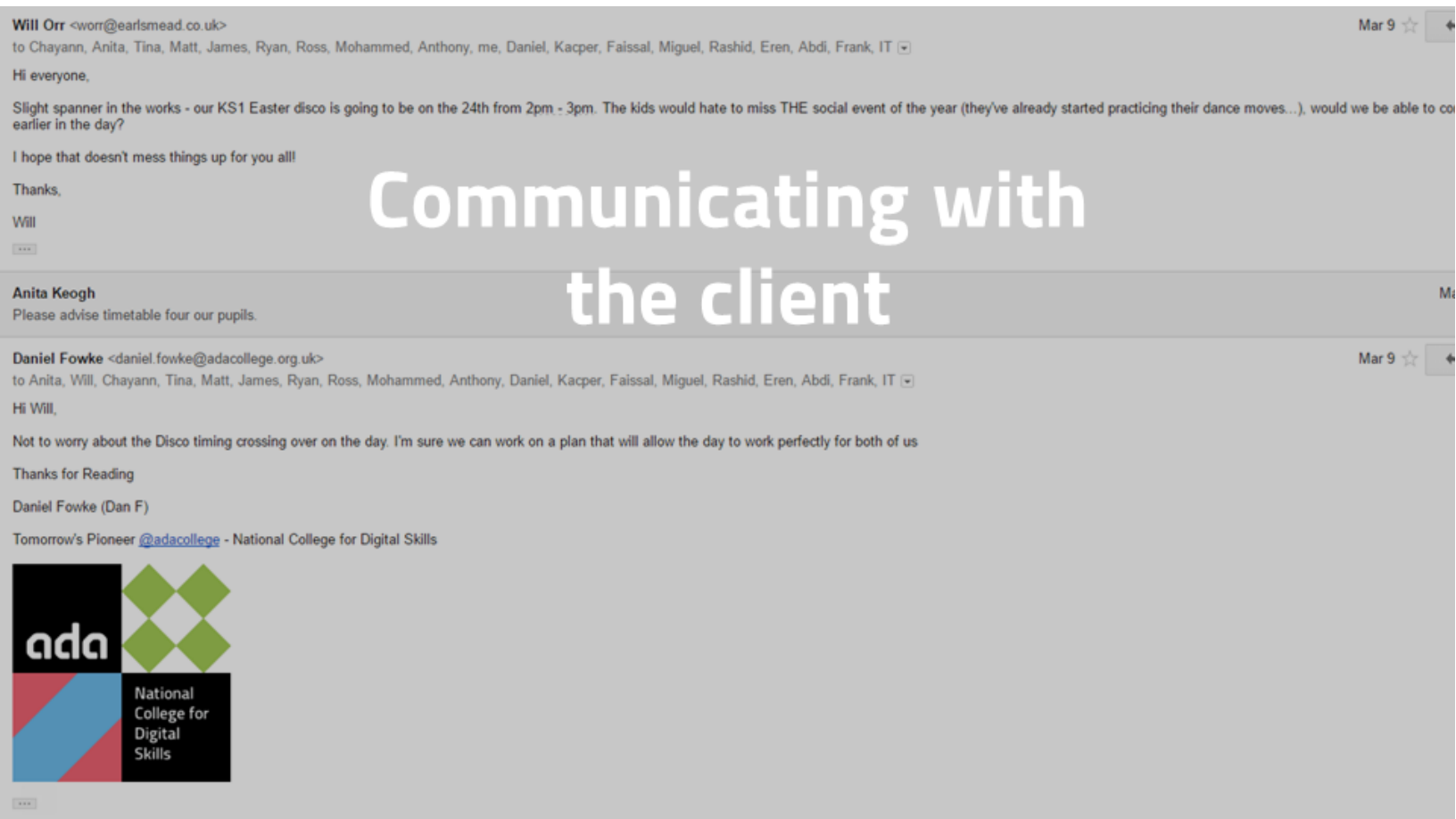
### Young

The target users will be young, ages in Primary school and above, my game would have to consider what the generation of users are fond with, some immediately outline components to this is that my game shouldn't be too complex, it shouldn't be boring and would need something interactive to keep the young audience engaged. Unique content gets younger audience excited

### Educational

The game need to keep in the zone of something that would be taught to a young audience, something such as the Solar System, but not the GCSE or A-level Grade information of the solar system, but the basics and information that they could use in school either in the present or the future. Meaning the idea and story would have to wrap around this subject.

Daniel Fowke



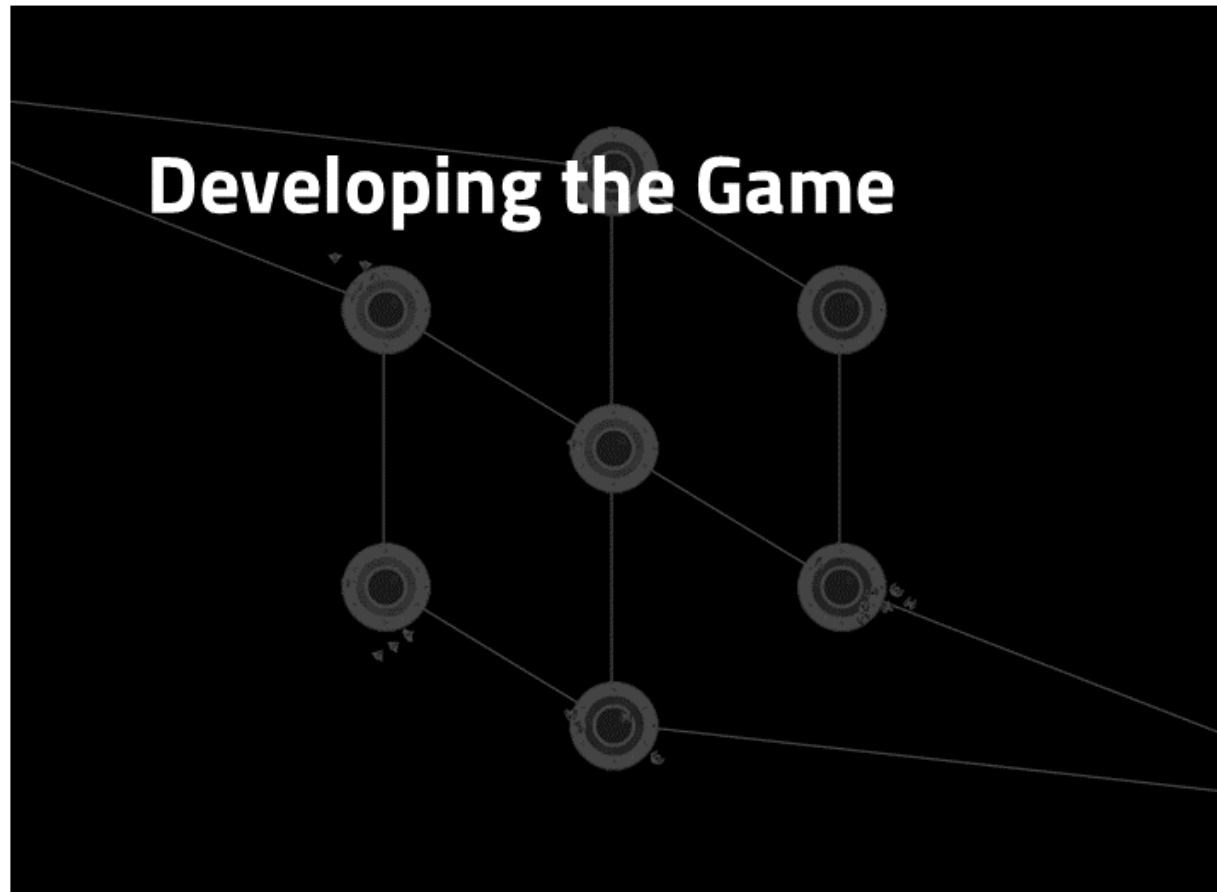
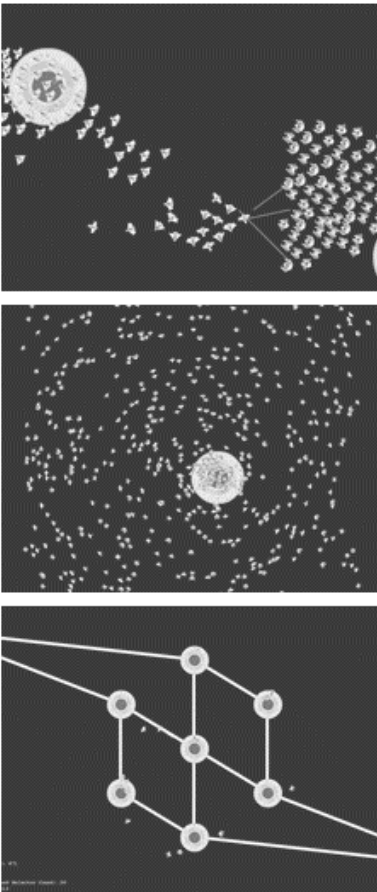
## Communicating with the client

During the development of the game I have taken note into the client requirements, the needs for that the game is targeted to primary school children, our clients were Earlsmead Primary School and Welbourne Primary School.

The client as well as us making the game for them would also be able to bring in a sample group that therefore I could test the game, this meant for me to work towards organizing the test day event for both schools to come in and test.

My communication with the client was resolving the issue that Earlsmead Students had an event on the planned the test day, this was possible to conflict with the test day. However to resolve this I informed them that this could be resolved and that the timings would suit both their and ours to make the test day happen.

Daniel Fowke



## Developing the Game

Following the Planning and timeline I created during its progress I made copies each time I reach a 'checkpoint' / milestone in the creation of the code. This allowed me to save each stage of the game to come back to if I ever need to refer or revert to an older version. When I completed a stage I marked it with the JS file that I completed this with, made a copy and proceed to test and complete the tasks.

The Final Test Build - <https://solo-hawk.github.io/Coursework/Game-Development/Before.html>

The Development Directory for referencing each build - <https://github.com/Solo-Hawk/Solo-Hawk.github.io/tree/master/Coursework/Game-Development>

Daniel Fowke

Date (Week Ending)	Objectives
3rd March	<ul style="list-style-type: none"><li>● Complete Timeline / Production Schedule</li><li>● Identify the assets to be used in the game</li><li>● Research BOID (Flocking Algorithm) and figure out what I need and don't need to apply it into my game</li></ul>
10th March	<ul style="list-style-type: none"><li>● Identify the limit to BOID count possible in PhaserIO and Web browsers without slowing the game the down (Logic to seeing this issue - the more individual BOID Objects the more calculations leading to more processing required per frame) - <i>test1.js</i></li><li>● Create BOID examples in Phaser</li><li>● Apply assets to BOID - <i>test1.js</i></li><li>● Create functions to make both sides attack each other - <i>test3.js</i></li><li>● Create death and respawn handling to the game - <i>test4.js</i></li><li>● Identify how to implement the methods of control from the user - <i>test6.js</i></li></ul>
17th March	<ul style="list-style-type: none"><li>● Create the first Level Map - <i>test8.js</i></li><li>● Complete the AI Development so the opposing side can move correctly following the same rules as the player - <i>test8.js</i> -&gt; <i>Corrected in test9.js</i></li><li>● Create random map generate - <i>Discarded, too complicated for time frame</i></li></ul>
24th March - Test Day	<ul style="list-style-type: none"><li>● Create Interaction Highlighting for selecting ships - <i>test11.js</i> -&gt; <i>Corrected in test12.js</i></li></ul>

Daniel Fowke



## Feedback from initial testing

### Feedback from initial testing

For the testing I had the access to talk to primary school kids that would meet my target audience, being that they played the game and passed my expectations in that most of the testers understood the game, and controls with only being explained to them that they can use the trackpad to the laptop

The feedback that I took was in a long series of post-it notes and a video recording. Since the response were very open they all could be condensed to come under just a couple of headings under certain aspects of the game and from that allowed me to create a table regarding certain parts of the game and what went well and what didn't

Test Day Video Evidence - <https://youtu.be/nCvJN71P6d8>

To this I simply flagged a tally for when a certain term was said;



Daniel Fowke

## Movement / Controls

Understood moving at the start	6	3
Difficulty with Controls		
Understood Travelling	3	
Tried Keyboard input first	2	
Doesn't understand rule of movement	2	

## Identification by user

Bugs / Birds / Creatures	3
Aliens	1
Control / Domination Game	4
Blue is Good, Red is Bad	3
Shooting	3
Circles are Bases	2

## Design

Background (Stars, More space-like)	4
Bigger Ships	1
Colours	2
Map create Shapes	1
Clear	2

## Replayability

Difficulty Level (Harder or Easier)	5
More Levels (More bases, points, bigger maps)	4
Win / Lose conditions	3

## Improvements

Two Player	2
------------	---

## Fun / Boring

Fun	2
Play again if improved upon	1
Boring	1

## Interface

Understood Interface Feedback	3
Used Drag Select	2
Need hints to understand	2

Daniel Fowke

Title

1

From this I could see that the game was very enjoyable for most of the users, though they liked the game there were things that they saw could do with improving, while some said to make a multiplayer mode some would want the game to be harder and others would just want there to be the possibility to win (This was something that I missed in my planning stage and notices only during testing)

I took this feedback and with the remaining time left I took to go and make some of the easier improvements to the game

- Makes a new map
- Make the ships bigger

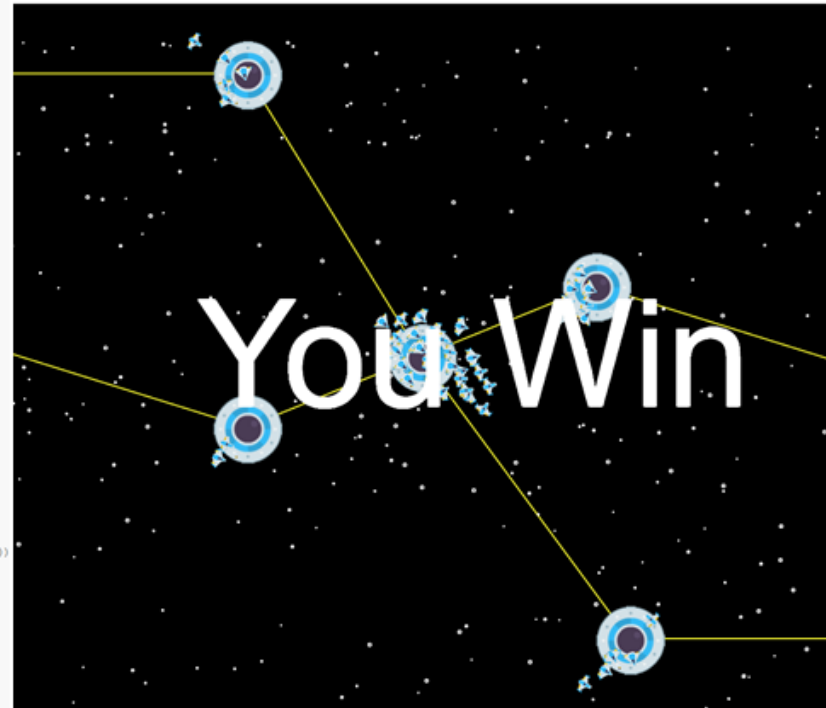
```

exists = true;
x = node[0].x + randX - 400;
y = node[0].y + randY - 400;

// deploying each hostile boid
dx = game.rnd.between(-50, 50);
dy = game.rnd.between(-50, 50);
action field for each ship
[0] = new Phaser.Circle(0, 0, boidRange);
ship creation
[0] = game.add.sprite(node[nodes - 1].x, node[nodes - 1].y, badShips[game.rnd.between(0, 2)])
[0].anchor.set(0.5);
[0].speed = boidSpeed;
[0].scale.setTo(boidSize, boidSize);
[0].force = boidForce;
[0].collideWorldBounds = true;
[0].physics.enable([0], Phaser.Physics.ARCADE);
[0].body.allowRotation = false;
[0].exists = false;
[0] = 0;
[0] = nodes - 1;
[0].kill();
[0].visible = true;
[0].exists = true;
[0].x = node[nodes - 1].x + randX + 400;
[0].y = node[nodes - 1].y + randY + 400;

// ship groups not collide
[0].arcade.collide([0], [0]);
[0].arcade.collide([0], [0]);
[0].input.pointerOver(0) {
  selectedNode = x
  console.log("5")
  console.log("pointer over : " + x + " x : " + node[x].input.pointerOver(0))
}

```



## Improving the Game

- Have a background

## Improving the Game

Making improvements to the game I make my next sprint in improving the game to;

Daniel Fowke

- Makes a new map  
This was done by replacing nodes, changing the connections between the nodes. What I did discover was how hard it would be to make multiple maps with different designs, my method of making the map wasn't the best and make it hard to make changes.
- Make the ships bigger  
A very simple fix that required me to change a variable to resize the ships
- Have a background  
Creating an image background with my artistic skill wouldn't result in something that even I would want to have as a background to the game, with this I ran to random functions and using the PhaserIO tools to make random white dots across the screen, a quick and hacky way of making a background. Though it is lacking in detail myself could see it have a bit more life to it, given the rather boring plain black background originally
- Able to win the game  
A condition to check the status of the node, once all nodes are Blue you win the game, though it does not stop or change the map once this is reached.

For being able to see the changes from my previous to my improvements I made a before the improvements and an after

Before - <https://solo-hawk.github.io/Coursework/Game-Development/Before.html>

After - <https://solo-hawk.github.io/Coursework/Game-Development/After.html>

Daniel Fowke

# Future Improvements

## Future Improvements

Reaching this stage of the project I noticed I missed something very important in the criteria that needed to be met. The idea of the game being educational is something that was not made in the current version(s) of my game, this is because my educational aspect of the game was not in the core gameplay but in the story and side-core of the game. My bigger picture Idea being that the mission map (the map that would select the next level from) would be the Sol Solar System and as you progress you get Primary - Lower Secondary School level information on the various planets such as Size, Distance and its unique facts.

Because of this the first thing I would do in future improvements is to complete the first level of the Core Game to then be able to make the Menu and outer game assets to where I could actually implement the educational part of my game.

Other than that I have a lot of other improvements I would want to make to my game in the future, based on my insight on me making the game and the feedback from the testers that I didn't cover;

- Optimize code, allowing to have more BOID ships on the screen at one time, this would be done by reducing loops and shifting code around
- Create a tutorial to the game

Daniel Fowke

This would be something that would require a lot of work, as it would require more event triggers than the actual game. Setting up assets just for the tutorial as well would be time consuming

- Create a ported version on desktop to support touchscreen  
Originally looking into touch screen during my development I uncovered that it would require to make a separate detection system for touch input as it actually has a unique input aside the mouse
- Difficulty Levels (Easy, Normal, Hard)  
Something that would require a menu and events that would change the difficulty based on the player's skill, to make the game a bit "Intelligent"
- Two Player  
To do this I would need to potentially redesign the game that would take some time so that the game would be optimized for 2 player usability across two separate devices
- Better Ship Highlighting  
My original highlighting system was done exploiting debugging functions from Phaser, though they work well I can do better

Daniel Fowke

# Reviewing and Conclusion

## Reviewing and Conclusion

### Reviewing computer game

The game was clear that it appealed to the target audience, through my first prototype idea to my first test run of the game on the Game test day, I got very positive results in the idea and how the game worked, the quality met their requirements in terms of playability and being able to play the game.

The game suited the target audience, though the visuals make them mistake the ships for insects on a few occasions all of them seemed to enjoy the game and not find the shooting aspect too violent that even the Teachers / Guardians seemed pleased of it, showing it was clear that I met my target audience expectations

Though my game up to this point didn't meet the requirements of being educational, my plans do make it clear that the core of the game isn't the educational part and that in future plans would the game have the educational aspect implemented. Otherwise I met the needs for the game being made for a young audience

Daniel Fowke

For ethical constraints all that I thought of during the design and development of the game is how to make the game involve shooting to be as non-violent as possible. As this is a young audience, this was where the idea of Spaceships, something fictional made sense and led to the space setting. Doing this clearly made the game acceptable to the target users and then most focused on the controlling the bases aspects rather than the ships shooting each other.

For the technology part of the game given that I had the game made for a web browser would I expect that the PC or device that the user would run the game from is not necessarily a high end Gaming PC or as such a High-end mobile phone. Meaning my constraints was not to drive the game so far that it wouldn't run on the lower end devices.

The strengths of the game was surely the simple click based controlling, this was how you moved in the game and what allowed the user to play the game. This is what was commented in the user testing as a good aspect of the game, making it more interesting.

As the game currently sits it only for the desktop platform through a web browser, if to future interactions I was to improve would only then I have made the game playable on mobile through web browser then to a stand-alone port into a phone app

## Quality characteristics

The game seems very comparable to other simple web games to the users and is what made my game fit right in while standing out. Though the playability and performance as par at most there was a clear possible need to make better improvements on those qualities

On testing the game with uses was the design very plain and bland, if in the future to present the game better I would have made more assets to add into the game and at least a Title Logo

## Skills, knowledge and behaviour

### Skills

My skills in effective, efficient and abstract programming improved over this course as with my code I went from 5 large for loops all the way down to 1 large for loop and 2 small ones, this was done all in the optimization stage of my first part of the prototype code. Something that I will be constantly improving but this course made a jump in that.

Furthermore for me, my ability to interact with the younger audience was put at hand, I do believe I handled the test day very well where we had to communicate with the under audience to learn about what they thought of my game to then make feedback for my improvements. As during my testing did I have my last

Daniel Fowke

group try to be a very “Destructive” one I did manage to control the situation and still get valid and useful feedback

## Knowledge

My knowledge in the using PhaserIO was greatly increased, I learned so much in utilizing the basic level of functions to make my game work and could repeat the use of what I learnt to make different games and even take time to improve this game over iterations of revising the code.

Additionally to make my game exist on the web I used GitHub’s GitHub pages to host my code and games, this in turn meant I took the time to learn the basics of pull, add, committing and then pushing on Git. This course made me much better at making changes and applying them to my GitHub repository which I can take with me into the professional industry under any computing topic in the most part, my next improvement in this would be to make much better use of branches

## Behaviour

My communications both through Email and front to front with the client both went very well, my personal evaluation is that I interacted with the client formally, though not keeping it too serious to kill any mood, the atmosphere being right for the young audience and the teachers.

For documenting my work I took to visual, pictures and footage, of course with the right permission and the assurance that what was filmed is just for revisions of the game code.

My code I haven’t introduced any crediting links to my code out of the sources and assets I based off / used, that was because in the iterations of the code I didn’t see a clear space to add this, I should’ve still have had a commented out at the top of the code to at least do it from the start. This is something I should do next time I develop the code and future code. This would apply very well in the future and the professional world.

My responses to feedback was handled very well, I acted very open the user’s opinion and in cases agreed with them, I was very caring in making sure I was acknowledging their feedback to them allow them to open up to more feedback and criticism. I see this as something very crucial in the professional world, you have to be able to see thing from all points of view and something I did take on in the development of my game.