Steven Wallace

DE0972: FINAL PROJECT

12013171

CONTENTS

INTRODUCTION	5
INITIAL IDEAS	6
CHOSEN IDEA	7
VISION	8
INITIAL RESEARCH	9
GAME CONTROLLERS & RHEUMETIOD ARTHRITIS	10
MOTION CONTROLS IN HEALTH	11
MOTION CONTROLS	12
HAND MOVEMENTS	14
EXERCISE YOUR FINGERS	15
GLOVE CONTROLLERS	16
CONCLUSION OF SECTION	18
DIFFERENCE BETWEEN 2D & 3D	19
WHAT MAKES A SUCCESSFUL GAME	21
COMPETITOR ANALYSIS	22
VIDEO GAMES AS AN ART FORM	25
THE STYLE OF "LOW POLY"	26
MOOD BOARD	27
EXPERIMENTING WITH LOW POLY	28
CONCLUSION OF SECTION	29
USER RESEARCH	30
PERSONAS	31
SCENARIOS	34
STORYBOARD	36
INTERVIEW	37
CONCLUSION OF SECTION	38

CONTENTS

DESIGN DEVELOPMENT	39
PLATFORM GAME	40
SKETCHES OF GAME ASSETS	41
CONCLUSION OF SECTION	43
GETTING STARTED WITH UNITY	44
GAME DEVELOPMENT	45
- MOVING PLATFORMS	48
- CHARACTER CONTROLLER	50
- CAMERA CONTROLLER	79
- BACKGROUND PARALLAX	82
- PARTICLE EFFECTS	84
- LEVEL MANAGER	86
- CHECKPOINTS	88
- HUD	90
- HEALTH & DAMAGE	95
- FLOATING TEXT	100
- PATHED PROJECTILE	107
- PLAYER PROJECTILE	112
- ENEMY AI	116
- INSTAKILL TRAPS	118
- HEALTH PACKS	119
- MOVEMENT CONSTRAINTS	122
- AUDIO	123
- ANIMATIONS (BASIC)	126
- ANIMATIONS (PLAYER)	127
- ANIMATION CONTROLLER	128
- PAUSING THE GAME	129
- MULTIPLE LEVELS	132
- CONCLUSION OF SECTION	136

CONTENTS

GLOVE CONTROLLER	137
CONCLUSION OF SECTION	151
PROTOTYPING	152
VERSION 1	153
VERSION 2	154
VERSION 3	155
VERSION 4	156
VERSION 5	157
CONCLUSION OF SECTION	158
TESTING	159
FIRST TEST	160
SECOND TEST	161
CONCLUSION OF SECTION	162
BRANDING	163
NAME GENERATION	164
LOGO DEVELOPMENT	165
PACKAGING	167
CONCLUSION OF SECTION	168
CONCEPT VIDEO	169
PLANNING	170
STORY BOARD	171
FINAL VIDEO	172
FINAL COMMENTS	173

INTRODUCTION

For my personal final project I was in two minds whether or not I was to carry on with my Ford Showroom app experience from Semester 1 or to create a new project that would suit me better and show off my skills in which I haven't yet used in this final year.

I would like to achieve many things with this project, with it being my last. I would like to implement what I have learned over the past 3 years into one big project. My chosen technologies, what ever they may be must demonstrate my skills in a new way so that my project is something unique.

Independently I need to show how I can plan, design, organize and manage my project so that a successful project will be the end result.

INITIAL IDEAS

1.

Carry on my Ford Showroom iPad application from Semester 1. If I were to carry on this project I would build the app so it would no longer be a concept, it would be a real working app. I could do this but all I would be doing is code, I think the end result would be great but i think the concept wraps it up nicely; as a concept.

2.

My second idea was to create a game. This game would either be 3D or 2D. I like the idea of creating a car track game, which uses my 3D from last semester so there would be a kind-of link to my projects. I this would prove to difficult due to the timescale I would opt for a 2D game, possibly a platformer. I like the idea of incorporating Arduino motion controls into this project also.

3.

The third idea I have in mind would be projection mapping; I have never created something to do with this technology so it would be fun to do something like that. I would develop it further if this was to be chosen as I don't really have any ideas of what i would project yet.

CHOSEN IDEA

PROBLEM / OPPORTUNITY

Many people play video games today. The game pad controllers are proven to cause cramp and this can lead to Arthritis. Not only that but people with developed Arthritis struggle to exercise daily. The opportunity I see is to create a game controller & a game that is fun to play but also helps exercise finger movement effectively.

SCOPE / AUDIENCE

The project would be aimed at people with Arthritis primarily, who struggle with movement in their hands and also if that user likes playing video games. However this could be used for any casual gamer as a new way to interact with a game.

TECHNOLOGY

I will be using Arduino to create the game controller then hopefully Unity 3D to develop the game once the controller is built. There may be the implementation of AppleScript as this is used to 'talk' to my laptop through Arduino and Unity so the game controller controls the object/ character in the game.

INNOVATION

This is innovative because there isn't really a game controller that helps Arthritis. There are motion controls but that is for full body movement, I am focusing on the joints of the hands for my project.

VALUE

The value would be for the Arthritis sufferers, they can exercise their joints without the need of excessive-ness, this would be casual exercise that focuses just on their finger mus-cles. This could also prove a value to gamers who experience cramp when using a standard game pad, as this is a lot more natural type of controller.

VISION

My vision is to create a game controller that users can benefit from as it reduces the amount of pain in the finger/ hand joints. I will also create a game that is fun to play so it would feel like the user is not really exercising, rather enjoying as casual past time.

VISION

" I WILL COMMIT MYSELF TO A PROJECT WHICH REDUCES THE EFFECTS OF ARTHRITIS THROUGH THE USE OF EXPERIENTIAL & GAMES DESIGN"

- Steven Wallace, 29th January 2015

INITIAL RESEARCH

A Starting Point

GAME CONTROLLERS & RHEUMETOID ARTHRITIS

The Sun Newspaper reported in 2011 that "kids as young as eight are suffering crippling arthritis-like pain from using consoles and phones."

"Stenosing Tenosynovitis", a painful condition in which a finger or thumb locks when it is bent (flexed) or straightened (extended). What this means is that your Tendon sheath narrows (stenosis), our Tendons are tough, fibrous cords that connect muscles to bones. Tendons must slide easily through their protective coverings (tendon sheaths).

The finger and thumb bones have tendons that are responsible for bending and straightening the fingers, the outer covering of the tendon becomes inflamed. The tendon swells because of the constriction, sometimes forming a nodule, and is no longer able to move smoothly through its sheath. As a result, a finger may lock in an upward position as the person tries to straighten it. The condition usually happens in the ring and middle fingers and is common in most people, typically over age 30. In infants and small children, the condition generally occurs in the thumb, they dub it the "Trigger finger" as it's more commonly known.

http://im.ziffdavisinternational.com/ign_de/screenshot/v/videospiele-machen-krank/videospiele-machen-krank_yw5y.jpg

MOTION CONTROLS IN HEALTH

The more I look into motion controls; I find a greater need for this type of interaction. I have found that motion controls can help people with joint pain, help them exercise muscles in a pain free way. I can link this so they can still have fun too!

Exercise is crucial for people with arthritis, it increases strength and flexibility, reduces your joint pain, and can help combat fatigue. Many think exercise is extensive but with motion controls it isn't.

The player does not need to run a marathon or swim as fast as an Olympic competitor to help reduce the symptoms of your arthritis, instead they can play a simple motion controlled game.

This moderate exercise can ease your pain and help you maintain a healthy lifestyle. When arthritis threatens to immobilize you, exercise keeps you moving.

Exercise can help you improve your health and fitness without hurting your joints. Along with your current treatment program, exercise can strengthen the muscles around your joints.

The motion controls can:

- Help you maintain bone strength

- Give you more strength and energy to get through the day & make it easier to get a good night's sleep pain free.

- Help you control your weight

- Make you feel better about yourself and improve your sense of well being

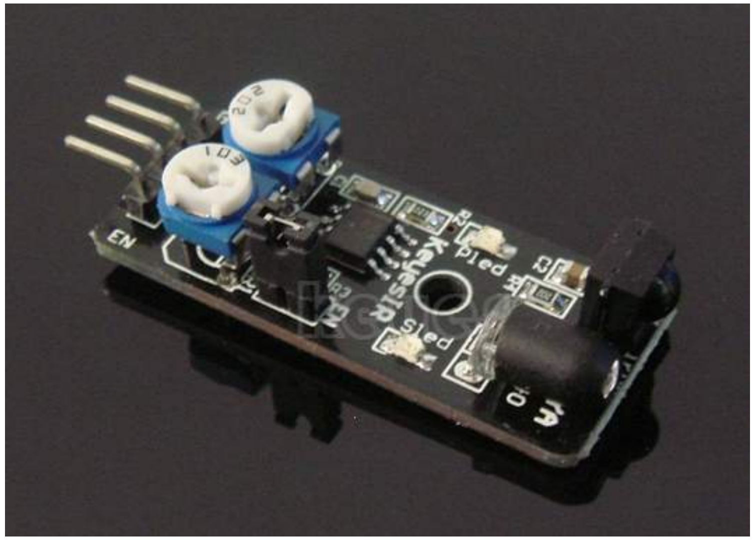
Though you might think exercise will aggravate your joint pain and stiffness, that's not the case. Lack of exercise actually can make your joints even more painful and stiff. That's because keeping your muscles and surrounding tissue strong is crucial to maintaining support for your bones. Not exercising weakens those supporting muscles, creating more stress on your joints.

MOTION CONTROLS

I think by including motion controls in my project I can make the traditional arcade machine feel as if it has a future, bringing modern technologies and making the overall gaming experience new and different to what new and old players have experienced before.

I have looked into IR sensors by using Arduino and linking this to Unity to control the character/ player. An infrared sensor is a device (usually with supporting circuitry) that can detect infrared light. Most of the remote controls for TVs and other entertainment equipment use infrared energy as the transmission medium to carry information between the control and the equipment to be operated. Infrared radiation is similar to light but has a longer wavelength, so we cannot see it without special equipment like a camera.

The KeyesIR Sensor is one that I have been looking at. This sensor is inexpensive costing only £2.50. This allows detection from objects in front of it. It is 4 pinned with GND, 5v, Out and EN. It also has an on-board LED, which is always useful, as I would not have to put my own on the breadboard. This can easily work with Arduino as it is built for it. The range of the sensor is around 30 - 40cm that would be ideal for a project that would be personal for the user as a game controller.

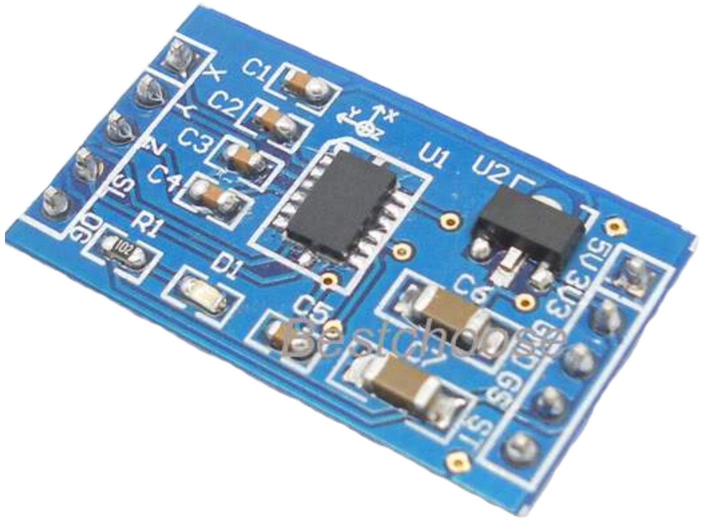


MOTION CONTROLS

The Triple Axis Accelerometer Module for Arduino is capable of gaming controls such as Tilt and Motion Sensing and has an Event Recorder, this could be ideal for me to use if I chose it as it great for motion controls, it is built for Arduino so would fit my project as it is capable of robotics, like motion sensing.

If i was to use this, it would have to strap it around the wrist some how, it is not huge but you would be able to tell there was weight on your wrist.

http://s1134.photobucket.com/user/onlyforever702/media/Arduino/



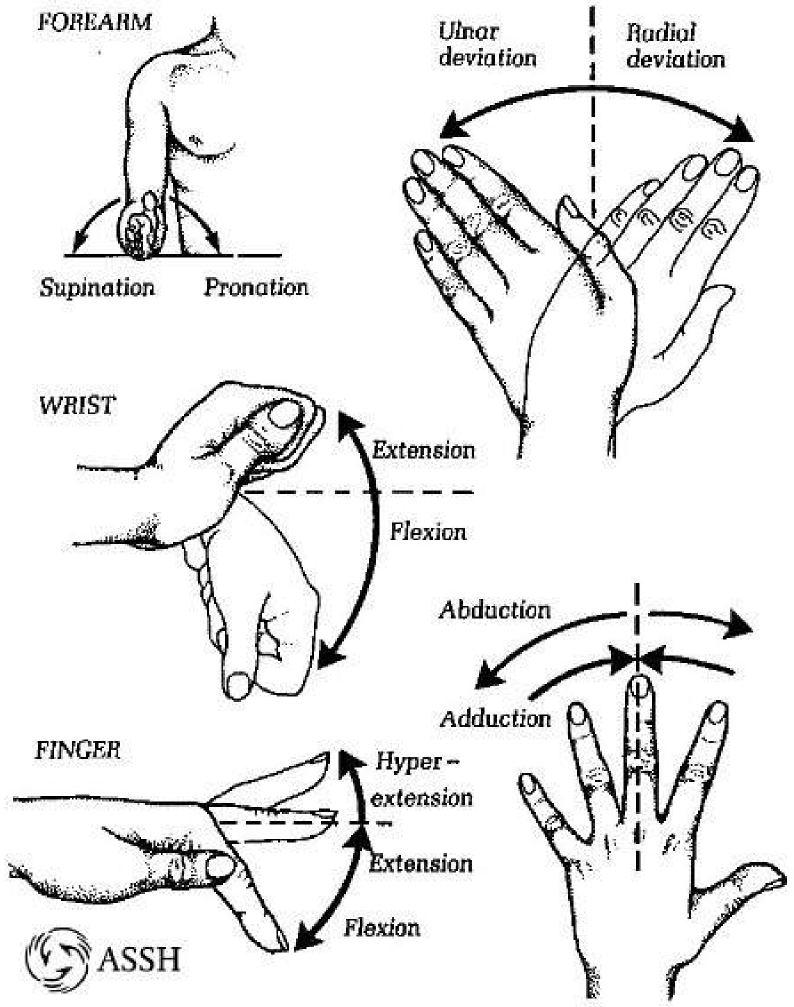
HAND MOVEMENT

As I look into motion controls further, I must look at the different hand movements we naturally make, in order to make the game controller feel as natural as possible and no strain is placed on the user from stubborn control movements, because the whole point of this controller will be to aid the user, not make the Arthritis worse.

I discovered this diagram to the left while researching hand movements; this gives me a clear indication of how to approach motion controls that are strapped to the hand. Of course this diagram displays the hand as a whole for most parts and I will be using fingers as a controller as well, not just wrist movements; what interests me the most is the finger diagram showing extensions (bottom left), this is perhaps the movement i need for my controller as it flex's the finger muscles.

Image:

http://www.handsurgery.com.sg/wordpress/wp-content/uploads/2011/01/hand_Movement1.jpg



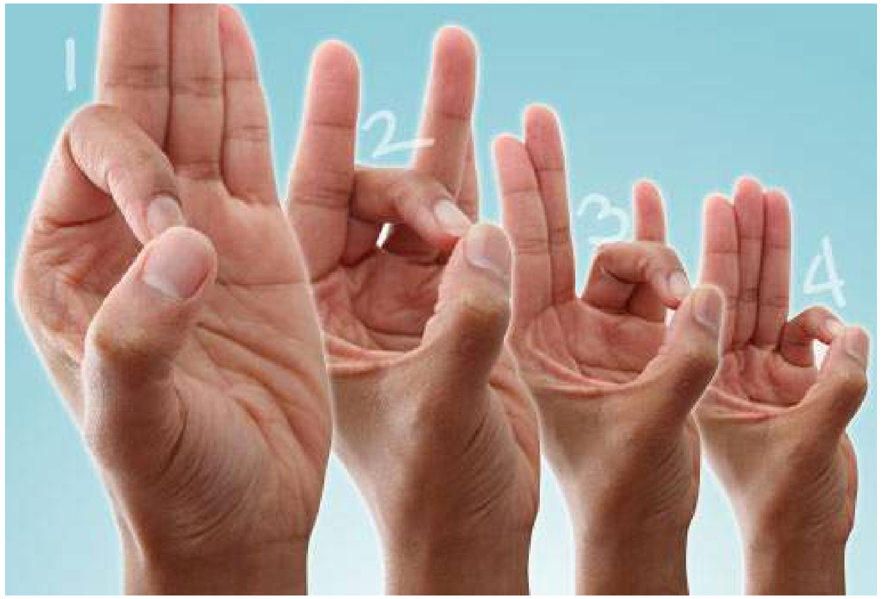
EXERCISE YOUR FINGERS

Thumb Touch

This exercise helps increase the range of motion in your thumbs, which helps with activities like picking up your toothbrush, fork and spoon, and pens when you write, an ideal exercise for Arthritis suffers as this is one of the main exercises recommended for Arthritis. This is a great gesture for a controller as each thumb touch could single a different control within the game.

Hold your hand out in front of you, with your wrist straight. Gently touch your thumb to each of your four fingertips, one at a time, making the shape of an “O.” Hold each stretch for 30 to 60 seconds. Repeat at least four times on each hand.

http://mydoctor.kaiserpermanente.org/ncal/Images/Thumb%20arthritis%20exercises_tcm28-181399.pdf



GLOVE CONTROLLERS

The Power Glove

The Power Glove is a controller accessory for the "Nintendo Entertainment System" of NES. The Power Glove was not popular and was criticized for its imprecise and difficult-to-use controls.

Originally released in 1989 this glove demonstrated how the finger flex measurement and hand position tracking using a pair of ultrasonic transmitters. This glove used optical flex sensors to measure finger bending, which is what I am focusing on for my glove.

The glove has traditional NES controller buttons on the forearm as well as a program button and buttons labeled 0-9.The user presses the program button and a numbered button to input commands, such as changing the firing rate of the A and B buttons. Along with the controller, the player can perform various hand motions to control a character on-screen. The general design makes the glove look futuristic and cool but the functionality just did not work, hence the poor sale of it and it is quite rare.

Image:

http://upload.wikimedia.org/wikipedia/commons/d/d3/NES-Power-Glove.jpg

GLOVE CONTROLLERS

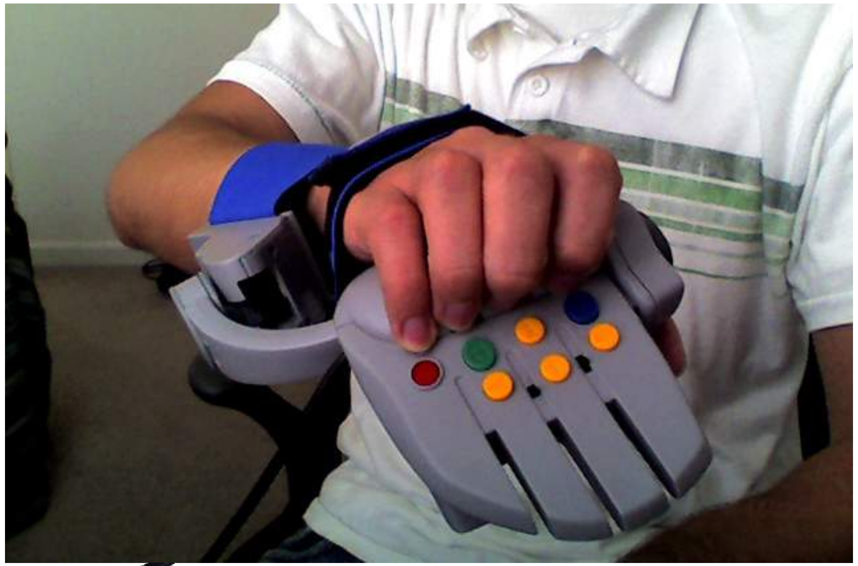
N64 Glove Controller

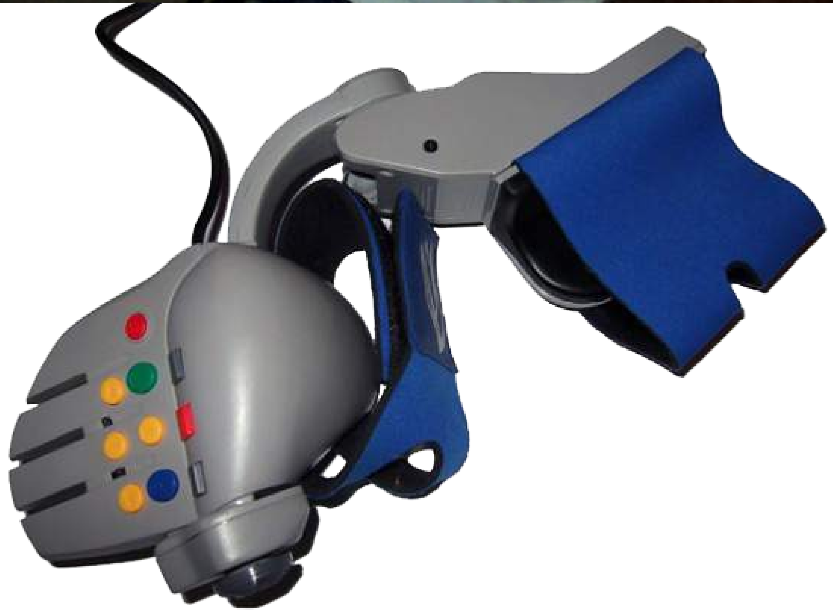
This is a controller for the Nintendo 64 game system. The controller wraps around your hand like a glove and has the joystick and face buttons within easy access. Like the Power Glove for the NES but it doesn't sense your movements, it looks a lot more complex than the Power Glove and that looked very complicated.

This controller wraps around your hand like a glove which is different to the power glove which you actually slip on, never the less it has the joystick and face buttons within easy access – it plugs into your console just like any other controller and can be used with any N64 game.

Images:

https://www.bing.com/images/search?q=n64%20





CONCLUSION OF SECTION

Looking over my initial research of how the controller could work, I am pleased with the progress I have made. I have identified that I want to make a motion-sensing glove, simpler to the ones that were on sale but flopped specifically for people who suffer from Arthritis and/ cramp.

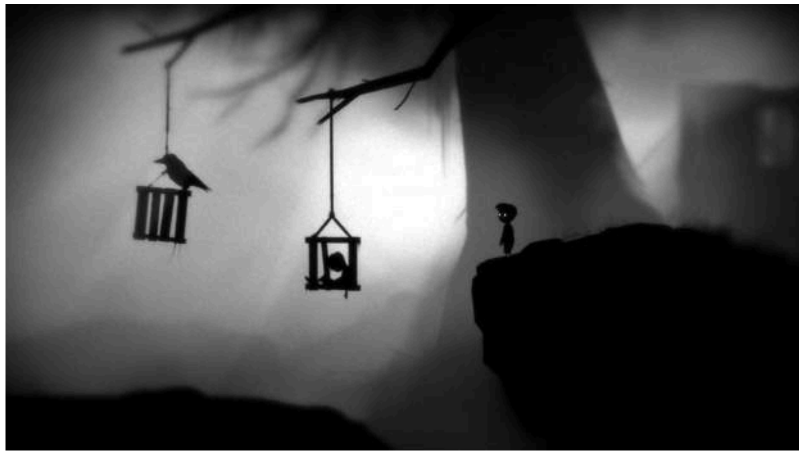
Looking into the hand and finger gestures we naturally make, I will have to design a glove around these factors so it functions as natural as possible so there is no strain on the user, after all, this is supposed to be mild exercise for painful joints.

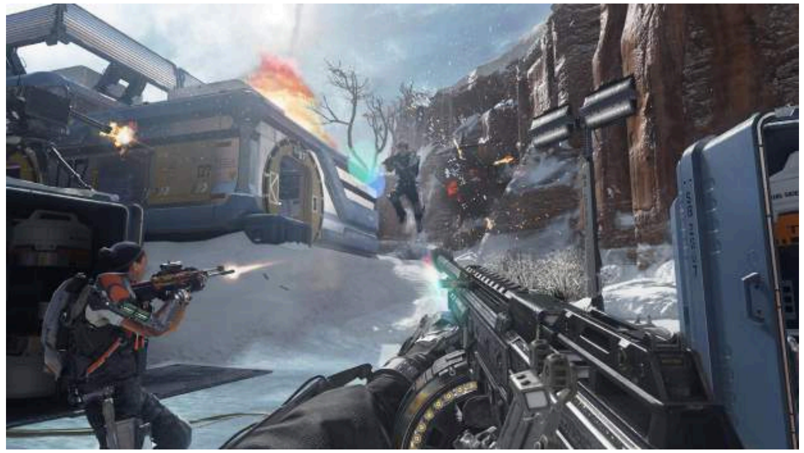
The next steps I will take are to look at the game type and style I want to put forward in my project.

DIFFERENCE BETWEEN 2D & 3D

Modern video games come in many varieties now. The major differences include 2D and 3D games. Many contemporary games focus on advanced 3D graphics, hoping to portray more realistic surroundings and characters.

However, game designers continue to produce 2D games. The choice to produce a 3D or 2D game also depends on the type of game, with many platforming games sticking with a 2D (see "Limbo" to the right) style and first-person shooters choosing 3D ("Call of Duty" Series).


http://db.tigsource.com/screenshot/image/1782/original/limbo2.jpg


https://www.bing.com/images/search?q=call%20of%20duty%20

DIFFERENCE BETWEEN 2D & 3D

There are 4 main factors that differentiate 2D & 3D:

Production

3D games rely on models, 3D shapes designed on the computer, whereas 2D games rely on sprites, 2D drawings on a flat surface. For example, 3D models can be compared to sculptures whereas 2D sprites are similar to flat drawings. Since it requires more art assets, 3D game production takes significantly more time than 2D game production. This is why I may opt for a 2D game due to the timescale I have for this project.

Visual Realism

One of the most apparent differences between 2D and 3D games is their graphical quality. Although 2D games oftentimes demonstrate excellent artistic design, 3D games are better at simulating reality like real world locations and faces. As a result, many games that strive for realistic visuals choose a 3D game engine but I feel like 2D is the way forward because it is a lot more 'casual'.

Movement

Movement in games depends on whether they are 3D or 2D. 3D games allow players to move in a 3D world, meaning that players can move closer and deeper into the screen. On the other hand, 2D games restrict player movement to a flat plane, usually left and right, but may include various other directions as well. For example, Rayman Jungle Run this game asks players to navigate a 2D world, moving from left to right until the goal is reached.

Controls

Since a game's dimensions determine the player's range of movement, 2D and 3D games use different ways to control their characters. 3D games use joysticks to control their avatars. Joysticks allow players to move around in a 3D space and are tilt sensitive to control a character's speed. On the other hand, 2D games use digital pads that allow players to press up, down, left and right. I am willing to change this to motion if possible.

WHAT MAKES A SUCCESSFUL GAME

"We also often discuss the desire for games to be art-for them to be puzzles with more than one right answer, puzzles that lend themselves to interpretation... both [art and game design] entail posing questions - tough ones even, ethical ones, even. And games will never be mature as long as the designers create them with complete answers to their own puzzles in mind."

Raph Koster 'A Theory of Fun,' Keynote Address to the Austin Game Conference, (2003)

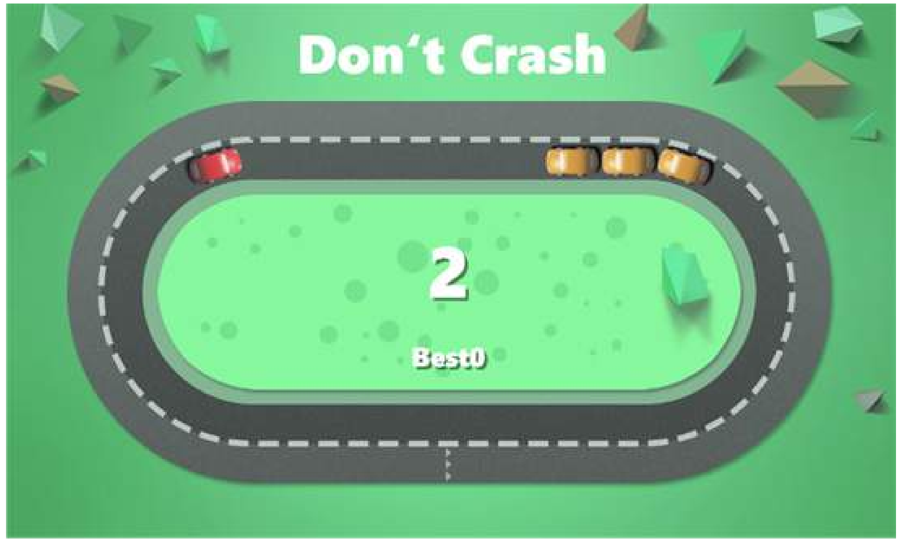
There needs to be a continuous challenge. A good game designer gives his players continuous challenges, each of which leads to another challenge, to keep them "hooked" on playing a game. This can be done by setting clear, short-term goals appropriate to the level of the player and the context within the game. I have the idea of a platformer with enemies or traps that offer a challenge to the player in order to progress, they must get past them. An interesting storyline is not necessary but can help, i guess that this will start to develop once i know what type of game i want.

There needs to some flexibility in the design of my game, each respawn, the track or platform changes possibly? This way the player cannot memorize the track and compete for the highest score; it is always a random selection of enemies/traps. Useful rewards could be useful, such as power ups or an extra life, this can keep the player playing the game for longer and feel more engaged in the game. Finally, the game needs to be fun, no one like a boring game so I will try to avoid this at all times!

COMPETITOR ANALYSIS

Don’t Crash!

This is a simple car game where all you have to do is not crash. Available on mobile/ tablet and PC. The player has to avoid the oncoming vehicles by changing lanes with a single tap anywhere on the screen, meaning the controls are very simple, something which i have to take into account. Don’t Crash is a simple game with not much narrative but works really well, I was intrigued by the graphic style of the game as it is low poly, and this is what i think I may look into doing with my game.

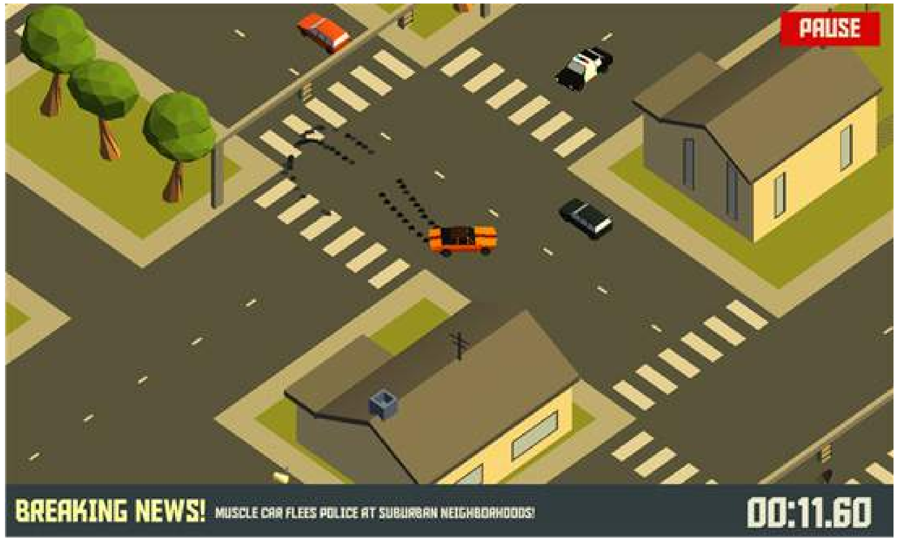


http://cdn.marketplaceimages.windowsphone.com/v8/images/8e6f8a63-ff78-48fb-93a3-a73534a9b6cc?imageType=ws_icon_large

COMPETITOR ANALYSIS

PAKO

This games tag line "Closed area. No Escape. How long would you last?" says it all really, it is a chase game where you have to avoid the enemy for as long as possible, meaning the police. This is a 2.5D Platformer/ Racer; it has the platform style, which I like but also this low poly look that I really do like. The controls are quite hard, it is not very responsive on mobile so i would have to make sure my controls are not as bad as this, but still make something look as nice as this.


PAUSE
BREAKING NEWS! MUSCLE CAR FLEES POLICE AT SUBURBAN NEIGHBORHOODS!
00:11.60

http://www.gamezebo.com/wp-content/uploads/2014/08/screen1136x1136144.jpeg

COMPETITOR ANALYSIS

RAYMAN JUNGLE RUN

Rayman is a legendary gaming icon, and as a Platforming hero to many I was wanting to look into how the latest game stacks up on mobile, like the other games, this is again mobile. No games I have found use motion controls so i know my project can be unique in that way. The games art style is beautiful, not my first choice but still looks great. The controls are the standard tap then tap again and I want to try steer clear of this. The Platformer is 2D, the simplest way of having an action game. I think this really works and like the other games I will have to see what they offer and make my project stand out.

http://smartcanucks.ca/wp-content/uploads/2014/12/Rayman-Jungle-Run-canada.jpg

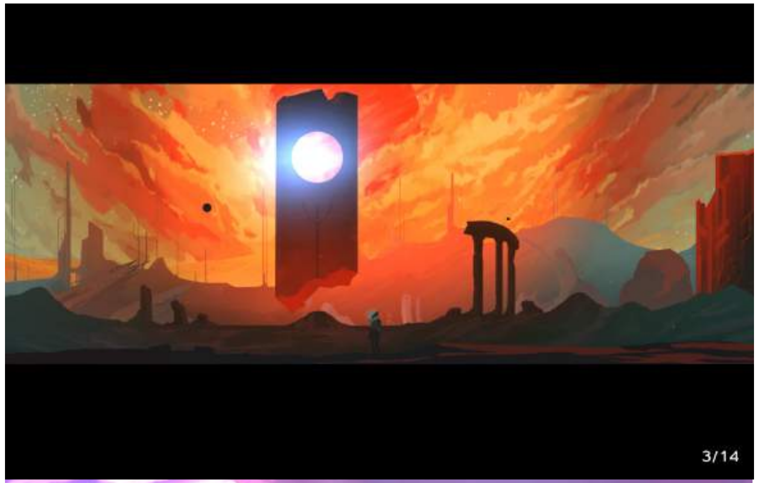
VIDEO GAMES AS AN ART FORM

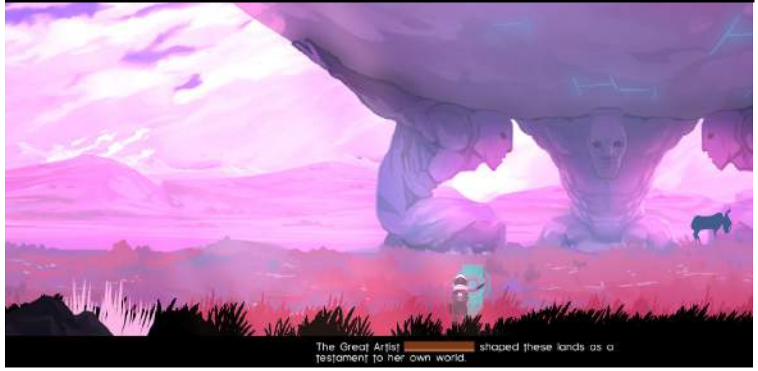
When I think of games, I don't just think about the story and game mechanics, I look at the art style and how every game is visually pleasing in their own unique way.

The concept of video games as a form of art is a controversial topic within the entertainment industry. The contribution of expressive elements such as graphics, storytelling and music can put forward a sense of art; I believe every game is purposely designed to be a work of a creative expression that the games designers are trying to portray.

A great example of art in games is "Elegy for a Dead World " this is a side-scrolling exploration game where the player writes a diary visible to other players. The player explores three worlds inspired by British romantic poets Shelley, Byron, and Keats. While exploring, the player makes notes on their observations.

Early critical reception has focused on the novelty of the note-taking mechanic and the emphasis on narrative storytelling over action, perhaps this is something I should consider...



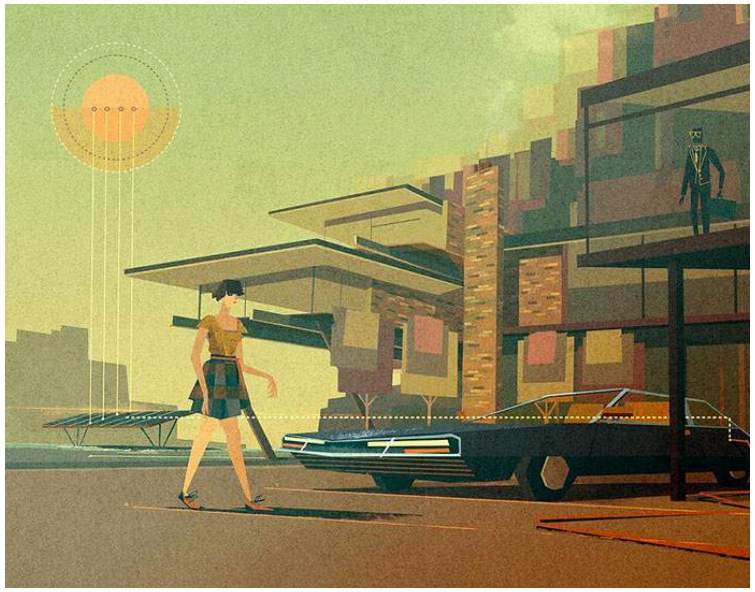

The Great Artist shaped these lands as a
testament to her own world.

"Elegy for a Dead World may be a jarring departure from traditions, but unlike the more pretentious attempts at alternative gaming experiences we've seen in recent times, this is one idea that should be encouraged." - *Hardcore Gamer*

https://www.bing.com/images/search?q=elegy+of+a+dead+world&FORM=HDRSC2

THE STYLE OF "LOW POLY"

I have noticed a common look in many images, not only in graphics and animations online, but also in magazines, and in television motion graphics, featuring low-detailed, faceted models, highly rendered, often with soft lighting effects. It's at once a reference to the early days of computer modeling and animation, but given a modern twist. This is the low-poly look. This is an ideal style for me as I want it to look retro but with a modern twist, I feel like this games design could really work.



Various images from Google Images.

MOOD BOARD

LOW POLY DESIGN

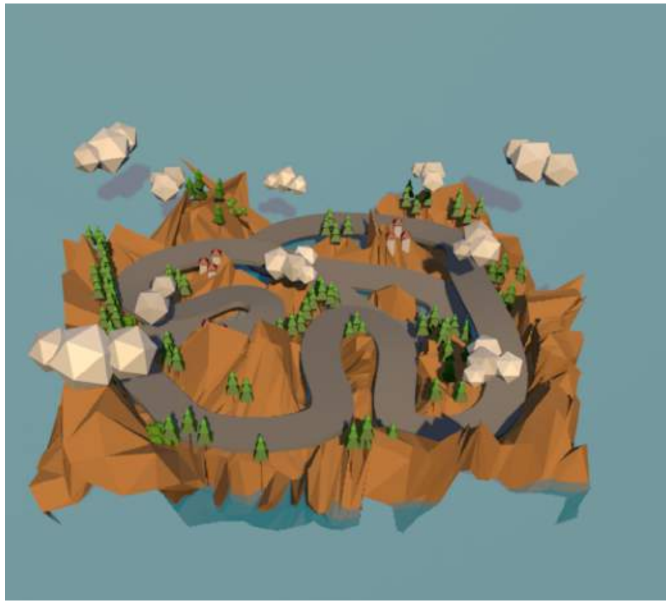
Various images from Google Images.

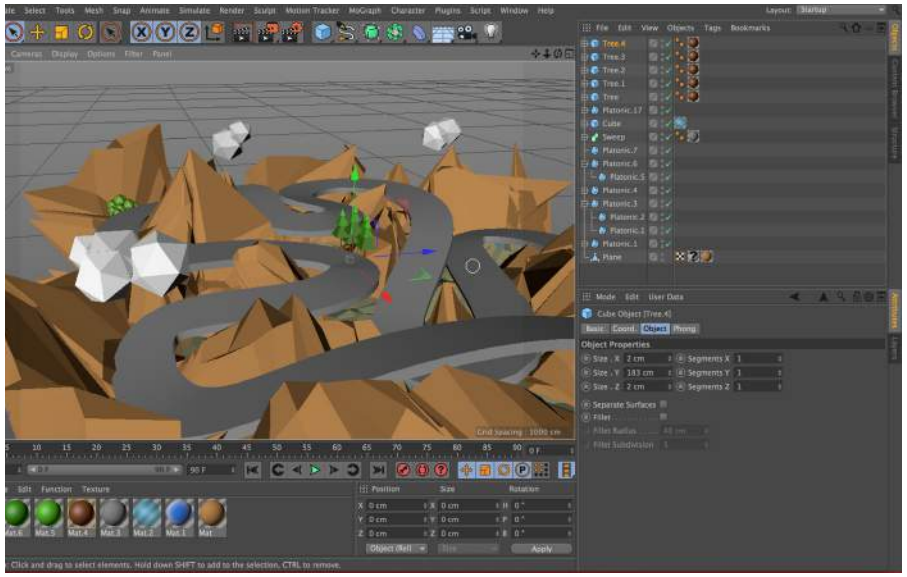
EXPERIMENTING WITH LOW POLY

Creating a low poly design in Cinema 4D.

I am intrigued by this low poly design so i really wanted to have a go early on. I know I'm no longer thinking of designing a 3D racing game as this would be a bit more difficult and the timescale is just not big enough, but never the less i have mastered the art of low poly!

I am really pleased with the way this has turned out, the style will definitely fit my game while still not steering to far away from the pixelated traditional look and not too close to the hi-def gaming that we see today.





CONCLUSION OF SECTION

After researching what type of game I want I think I am going to go down the 2D platformer route. I would love to do a 3D game but I think creating a controller as well as a 3D would be too much and I would not get it finished, I need to use my time effectively and I feel if I polish a 2D game to a high enough standard I will be really happy with the results.

Low poly design is the style I am going to take for my game as I think that would really make my game stand out as there are not many of these style games. I need to develop my game ideas so it can be fun to play along with the controller, but next I will look into how this would benefit the users of my project.

USER RESEARCH

Who am I designing for?

"I always get cramp off the controller"

JAKE **KING**

GAMES DESIGN STUDENT

ABOUT

AGE: 20
OCCUPATION: STUDENT
LOCATION: NEWCASTLE UPON TYNE
STATUS: SINGLE

Jake is a 3rd year student at Newcastle University, studying BA (Hons) Games Design. He loves gaming, its his life, he was first inspired by the retro 1980's arcade machines at the local fair.

Since his education he has learned to develop his own games and he feels the tools on offer today are amazing to create something with ease, he believes that one day that physical game controllers that we see today will become none existent in favour of motion/ voice controls, a more advanced version of Xbox Kinect is the future as the current controllers hurt his hands!

Image used: https://www.bing.com/images/search?q=student%20phot%20against%20wall&qs=n&form=QBIR&pq=student%20phot%20against%20wall&sc=0-13&sp=-1&sk=&ghc=1

GOALS

- Gain recognition for his games design
- Take a further leap into the digital world and connect to the internet of things.

INTERNET USAGE

- Gaming experience:	15 Years
- Primary uses:	Social/ gaming
- Favourite sites:	Steam
- Hours online per day:	8+
- Computer:	PC

"Who even likes educational games?!"

DAVID **SMITH**

COMPUTER SCIENTIST

ABOUT

AGE: 43
OCCUPATION: COMPUTER SCIENTIST
LOCATION: DURHAM
STATUS: MARRIED

David has his own computer-based business where he works with companies such as BBC and Unilever. When he was growing up arcade games inspired him to get into the computer based industry as his main hobby was going to the arcade machine at his local shop to play Pac-Man when he could afford it.

Sometimes David wished that he carried on his passion for games and turn it into his career, but he cant change that now. He plays the PlayStation 3 and loves the motion controls in it. He loves fun games, the educational ones are so boring!

GOALS

- Go to Australia, he has always wanted to go
- Hand his business down to his son

INTERNET USAGE

- Gaming experience:	20 Years
- Primary uses:	Work
- Favourite sites:	Dribbble
- Hours online per day:	10+
- Computer:	Mac & PC

Image used: http://www.illinoisclinicalpsychologist.com/images/30%20year%20old%20man.jpg

"Arthritis affects me, games can help"

DUNCAN **BROOKES**

RETIRED

ABOUT

AGE: 67
OCCUPATION: RETIRED
LOCATION: DUNSTON
STATUS: MARRIED

Duncan used to be a Joiner; he is now retired and suffers from Arthritis. This affects the movement in his hands and hurts him a lot. Daily exercise do help cope with the pain but he found that playing a games console while looking after his grandson, exercised his fingers a lot and felt better.

Since this discovery Duncan has become a lot happier in himself, he feels less joint pain and gets to spend time with his Grandson, who also loves games; which is a bones.

GOALS

- Cure his hand pain
- Take a further leap into the digital world get on Facebook!

INTERNET USAGE

- Gaming experience:	2 Years
- Primary uses:	Gaming
- Favourite sites:	Xbox
- Hours online per day:	2+
- Computer:	PC

Image used: http://mimg.ugo.com/201007/53628/cuts/funny-old-gamer_288x216.jpg

SCENARIO

Jake loves playing video games; it is his favorite past time. Yesterday he was playing and he got a sudden sharp pain in both of his hands and was unable to carry on playing, he had to turn the console off and massage his hands until slightly better.

The next day Jake goes to play on his console again and shortly after around 40 minutes of playing his game the same sudden sharp pain comes back and his wrist locks up. He cannot do a thing with it so decided not to play on it for a while.

While resting Jake knows it would be easier not to play the console but he is very bored and doesn't know what to do. He really wants to play on his game but cant, the controller is hurting his hands too much, and he can't get what he wants.

SCENARIO

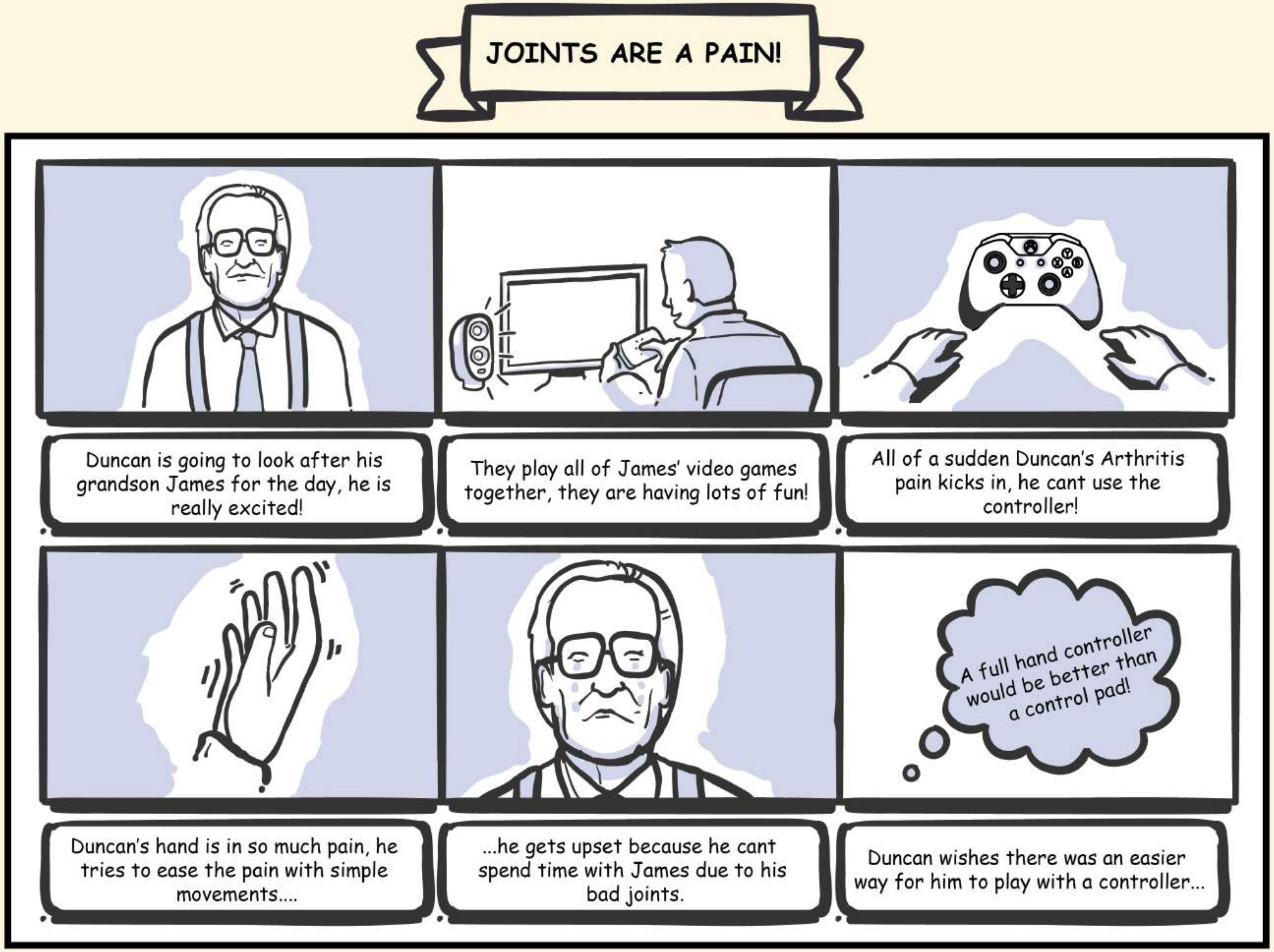
Duncan has been living with Arthritis for most of his life. He first started to notice swelling around his knuckles that increased the size dramatically over night and did not know what was happening until the doctor diagnosed him.

Duncan has daily exercises that he needs to do in order for the stiffness not to persist and live a little less painful life. However, As he's in his late 80's he often forgets to do these exercises as they are not always on his mind, he says they are boring and he has no interest in doing them.

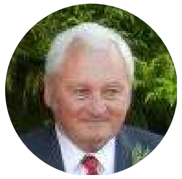
He has to wear a glove to bed that holds his hands in place so they are less painful in the morning when he wakes up, he likes wearing the glove because it makes his hand feel secure and he can grip better throughout the day.

He often wonders about ways to make the exercises enjoyable but can never think of a way, so he never does them, which he knows is bad.

STORY BOARD

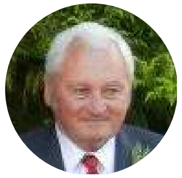

JOINTS ARE A PAIN!
Duncan is going to look after his grandson James for the day, he is really excited!
They play all of James' video games together, they are having lots of fun!
All of a sudden Duncan's Arthritis pain kicks in, he cant use the controller!
Duncan's hand is in so much pain, he tries to ease the pain with simple movements....
...he gets upset because he cant spend time with James due to his bad joints.
A full hand controller would be better than a control pad!
Duncan wishes there was an easier way for him to play with a controller...

INTERVIEW WITH MY GRANDAD



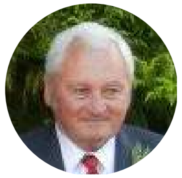
Hi Steven what would you like to know?

What does Arthritis feel like in your hand?



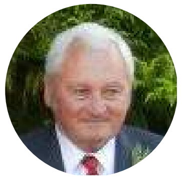
If you try to make a fist it feels as though you are trying to close your hand while you are clutching a tennis ball. The hand will not close completely.

How often does this pain happen to you?



Most of the time Steven, especially when you use your hands all the time, it is always painful.

How can I help you?



When you told me about a glove that can help with Arthritis I was very taken on board by the idea, there is not really anything out there that motivates us to do the exercises we are given to do. Something fun would work well, so its like we dont even know we are doing the exercises.

CONCLUSION OF SECTION

From the user research I have gathered I feel as if i have found a definite need for my project.

I need to create a fun game, not an educational one because no one likes them as they are boring and quite 'single pathed'. I need to make the user feel like they are just having fun, and not even exercising their muscles.

My support from my Grandads interview has shown me that there is a need for my idea and it would work and benefit the users.

My next steps are to finally start making my project. I will start off building the game first in Unity and learn C# as it is something I am eager to get to grips with.

Then, once I have a stable enough build I will take my attention to the glove controller.

DESIGN DEVELOPMENT

Lets make the game first

PLATFORM GAME

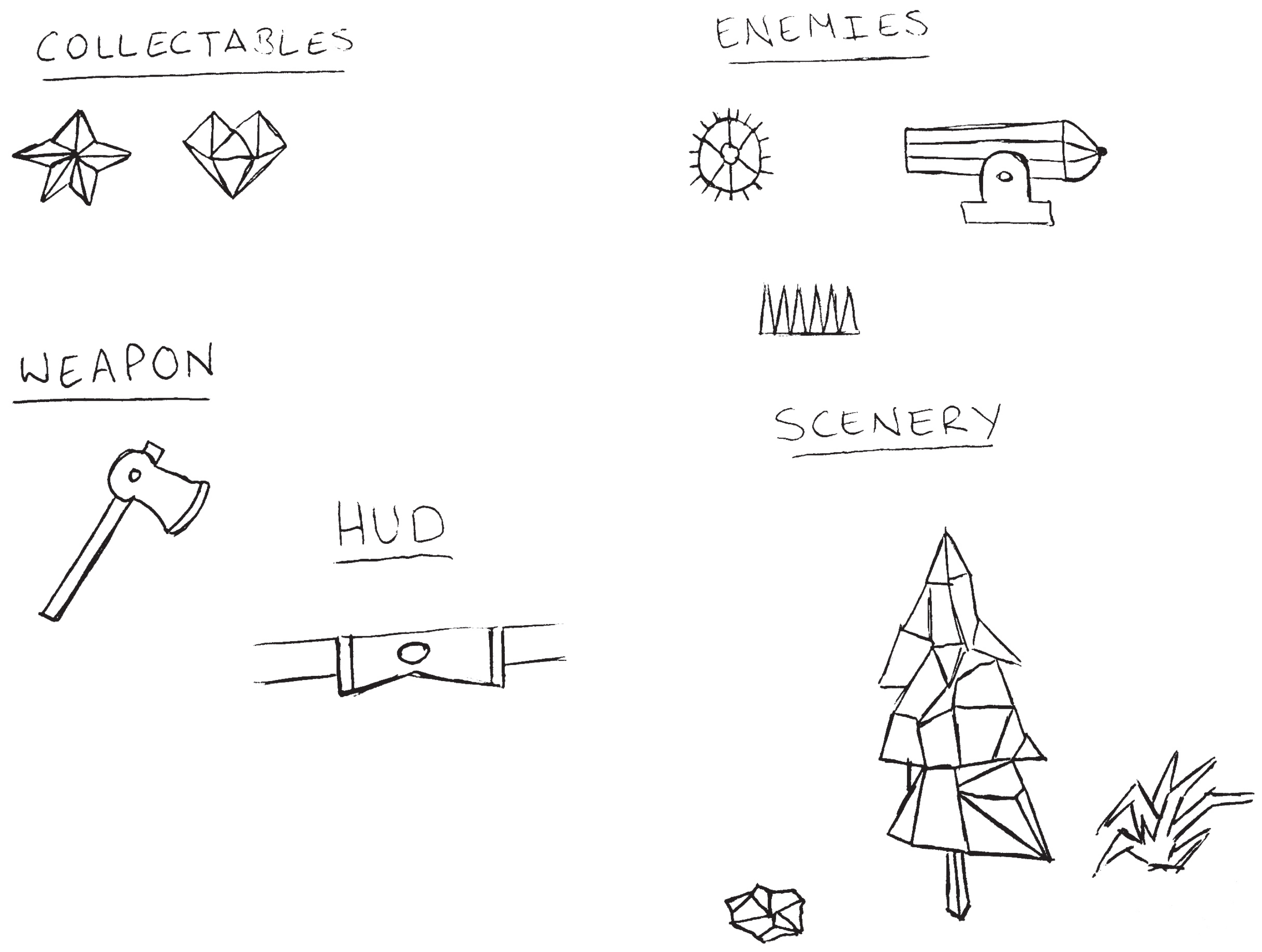
From my initial ideas i am going to create a simple 2D platform game to accompany my glove controller. The first step i need to take is to design and develop the game using Adobe Illustrator, Photoshop and Unity 3D software. Throughout this development i need to sketch out my assets for the game, then make a digital version so they act as a texture in unity.

The style of game I have already defined, low poly. However i need to place an engaging and fun scenario into the game in order for there to be a reason for it I am staying away from educational games as they have been proven to be not very engaging.

My initial thoughts are to have a knight, running through the outskirts of the castle, to rescue the princess as he dodges cannons, dragons and spikes. I need to sketch out my ideas in order to make my ideas come to life.

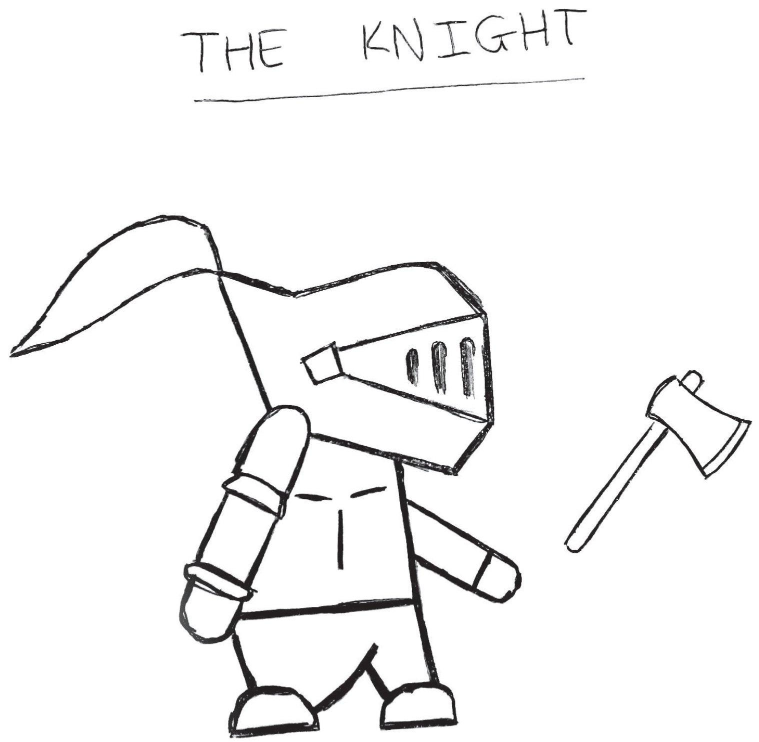
SKETCHES OF GAME ASSETS

Initially sketching out some quick ideas. Star collectibles, health packs, some of the enemies and scenery in the game are what will make up my game world, this is how I imagine the low poly look even though you cant really see it here in skeleton form.



SKETCHES OF GAME ASSETS

The main character, the knight. Again in skeleton form so the low poly look is not really visible here (I can do this in Illustrator) But this is how I imagine him to look.



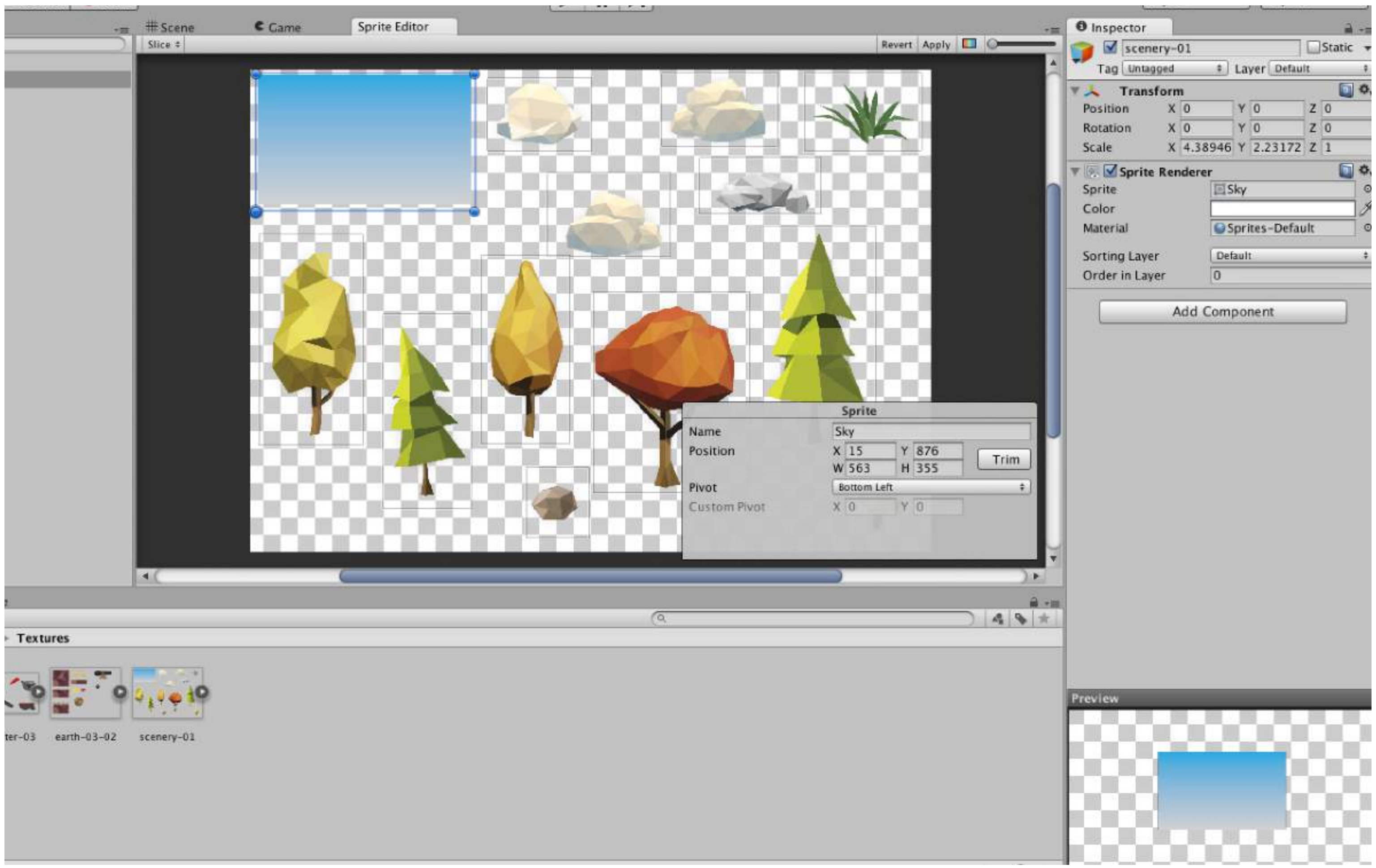
CONCLUSION OF SECTION

I have sketched out what I want in my game and have identified the style I want to use. The narrative that my game will follow is there and I have made those sketches into vector images that I can now use in Unity as a texture I feel like I have the basic objects to create a level. Later on I might decide to add other elements to my game for a little more complexity but for now I am ready to get stuck into Unity and create my game.

GETTING STARTED WITH UNITY

I will create my game first, and then focus my attention on the controller after. The first thing I need to do is create a new project in Unity, then import my hand drawn vector graphics as a sprite sheet so I have them as assets in Unity.

When I have imported the sprite sheet, I need to go into the sprite editor and manually select each graphic and assign them as an individual sprite, because at the moment Unity thinks that the whole sheet is one sprite. I could have imported each graphic as a separate sprite but this will increase the load time of the game which is not what I want. This is the simplest way.



Sprite
Name: Sky
Position: X 15, Y 876, W 563, H 355 (Trim)
Pivot: Bottom Left
Custom Pivot: X 0, Y 0

Inspector
scenery-01 ☐ Static
Tag: Untagged Layer: Default

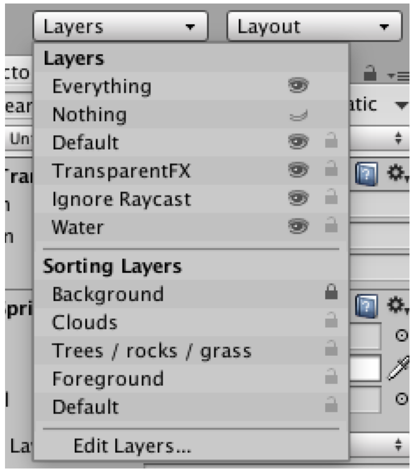
Transform
Position: X 0, Y 0, Z 0
Rotation: X 0, Y 0, Z 0
Scale: X 4.38946, Y 2.23172, Z 1

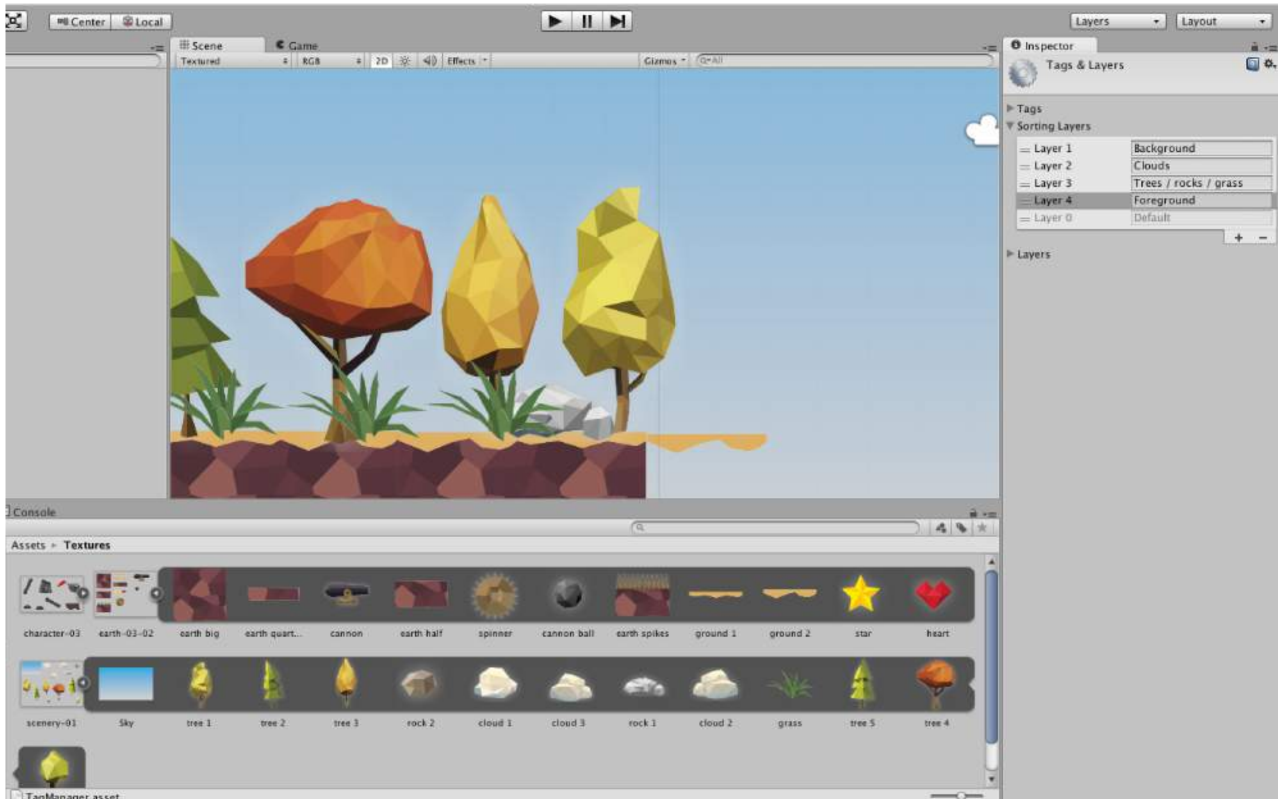
Sprite Renderer
Sprite: Sky
Color:
Material: Sprites-Default
Sorting Layer: Default
Order in Layer: 0

Once I have made clear to unity about my individual sprites I can start to place them in my scene view.

As I place them i need to sort out my layers so that no every object is the same, I need one for the sky, this will be the furthest away, then the clouds, then the trees, rocks and grass, finally followed by the foreground in which my platforms will be on. At the moment i am getting the world set up so I can then start coding.

I also need to lock the background (sky) so that this cannot be moved if I accidently click on it when editing the game world.

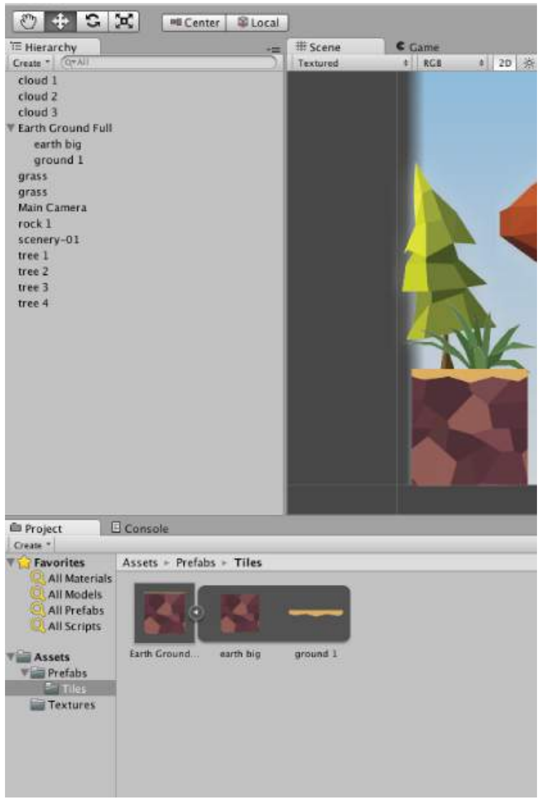


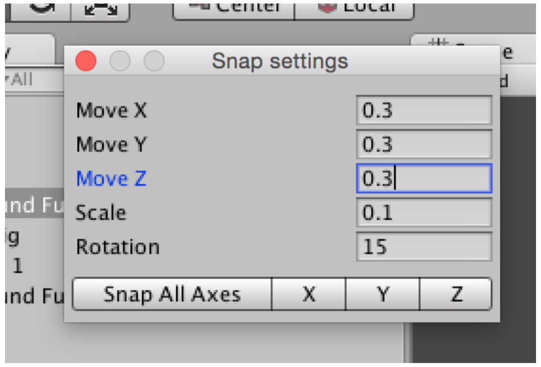


Layer 1	Background
Layer 2	Clouds
Layer 3	Trees / rocks / grass
Layer 4	Foreground
Layer 0	Default

When I imported my assets the platform ground had many sizes, I want to combine some of my sprites together and create a prefab. Prefabs are new game objects that I can make by parenting a child object (in this case the ground and dirt sprites) and saving them as a new combined game object, this will make things easier for me when I come to duplicate the platform.

Once this prefab was created, I was able to define the snap settings for it. These setting are used when you move and object and it snaps into place, making level design easier and neater.

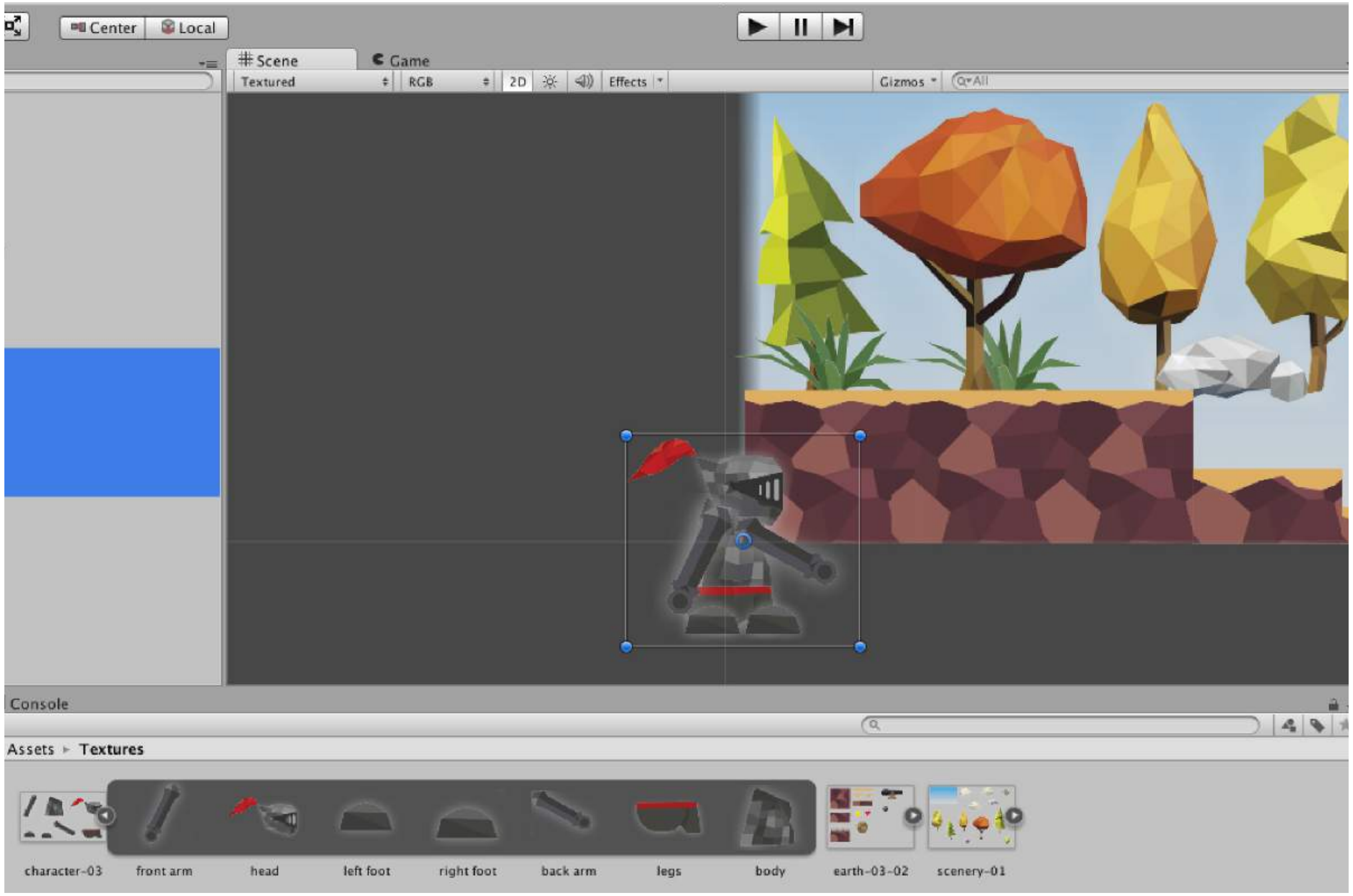



Snap settings

Move X	0.3
Move Y	0.3
Move Z	0.3
Scale	0.1
Rotation	15

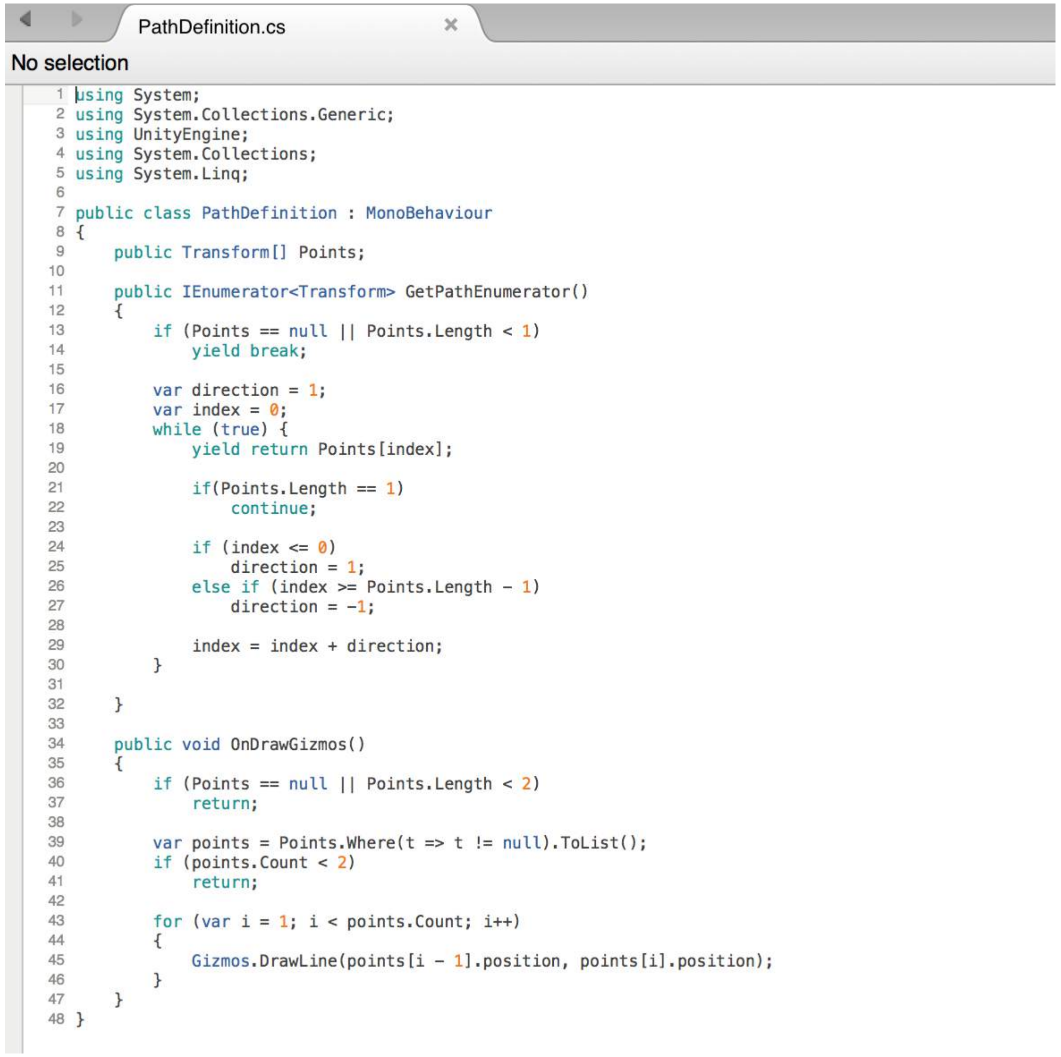
Like with the earth prefab I have just created, I assembled the player character together and selected all parts and created a new game object called player, I then defined the player to the sorting layer of "player" so that the character was on a layer of its own away from the scenery and platforms.

This is all I need yet for a very basic level design. The scene is set up so I get a feel of how it will look. The first game element I will focus on next are making the platforms move.



- MOVING PLATFORMS

For my moving platforms to work, I need to code a simple script, which tells Unity to draw a line between the points of movement that the platform will move from. This line will make it easier for me visually as I will be able to be the exact path of my platform in the scene builder window.

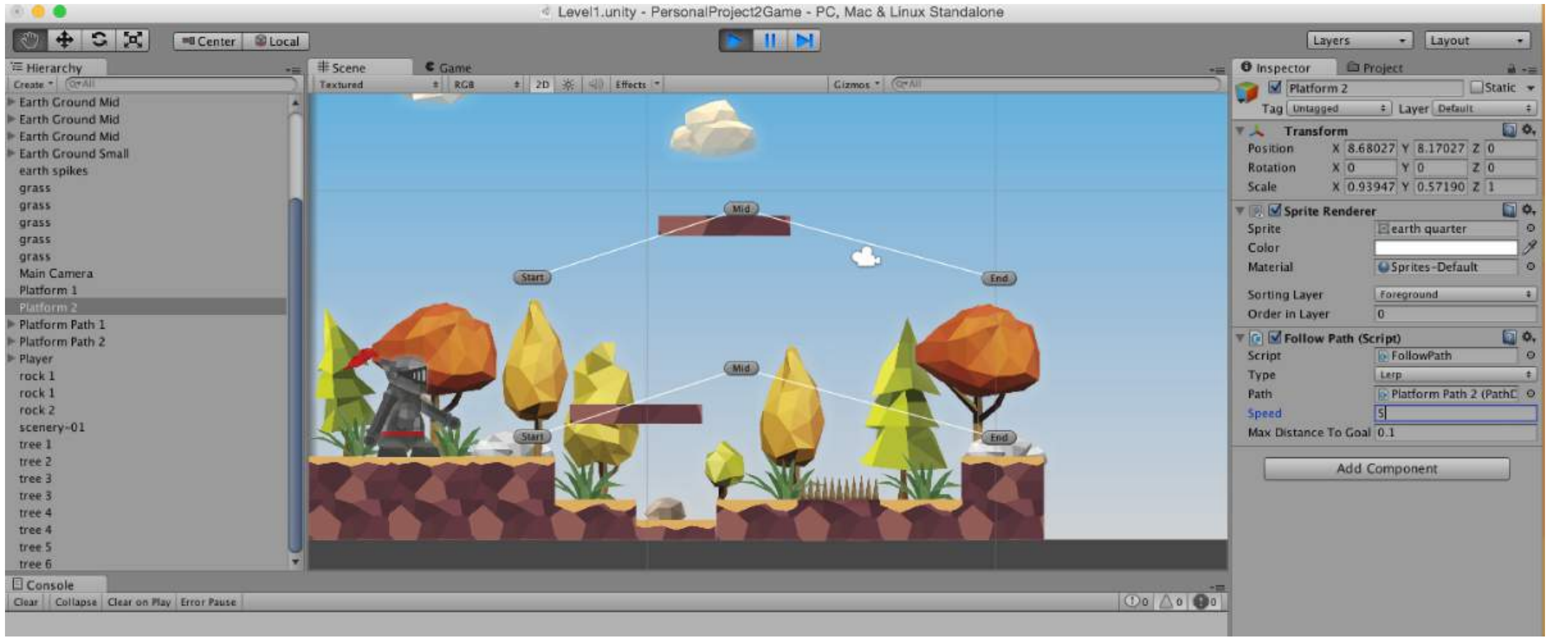

PathDefinition.cs

No selection

```
using System;
using System.Collections.Generic;
using UnityEngine;
using System.Collections;
using System.Linq;

public class PathDefinition : MonoBehaviour
{
    public Transform[] Points;

    public IEnumerator<Transform> GetPathEnumerator()
    {
        if (Points == null || Points.Length < 1)
            yield break;

        var direction = 1;
        var index = 0;
        while (true) {
            yield return Points[index];

            if(Points.Length == 1)
                continue;

            if (index <= 0)
                direction = 1;
            else if (index >= Points.Length - 1)
                direction = -1;

            index = index + direction;
        }

    }

    public void OnDrawGizmos()
    {
        if (Points == null || Points.Length < 2)
            return;

        var points = Points.Where(t => t != null).ToList();
        if (points.Count < 2)
            return;

        for (var i = 1; i < points.Count; i++)
        {
            Gizmos.DrawLine(points[i - 1].position, points[i].position);
        }
    }
}
```

- MOVING PLATFORMS

Once that code has been implemented, I created a new game object called "Platform Path 1", its child objects are "start", "mid" and "end", these are each other points for the moving platform to follow. On the parent object, I add the Path Definition script to it and here I added the start, mid and end objects to the elements.


Level1.unity - PersonalProject2Game - PC, Mac & Linux Standalone

Hierarchy
Earth Ground Mid
Earth Ground Mid
Earth Ground Mid
Earth Ground Small
earth spikes
grass
grass
grass
grass
grass
Main Camera
Platform 1
Platform 2
Platform Path 1
Platform Path 2
Player
rock 1
rock 1
rock 2
scenery-01
tree 1
tree 2
tree 3
tree 3
tree 4
tree 4
tree 5
tree 6

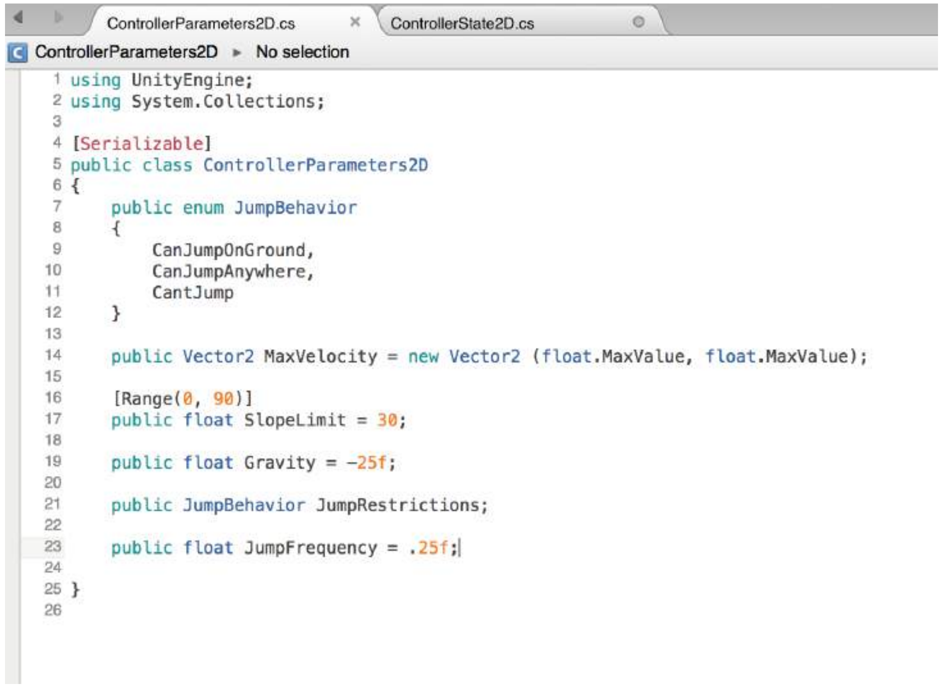
Inspector
Platform 2 Static
Tag Untagged Layer Default
Transform
Position X 8.68027 Y 8.17027 Z 0
Rotation X 0 Y 0 Z 0
Scale X 0.93947 Y 0.57190 Z 1
Sprite Renderer
Sprite earth quarter
Color
Material Sprites-Default
Sorting Layer Foreground
Order in Layer 0
Follow Path (Script)
Script FollowPath
Type Lerp
Path Platform Path 2 (PathD
Speed 5
Max Distance To Goal 0.1
Add Component

GAME DEVELOPMENT

- CHARACTER CONTROLLER

The character controller is responsible for moving the character around the world, perhaps the most important sections of code I need for my game.

I will write my own custom physics that allow me to control the gravity, velocity and collision classes. In the screen shot to the right I am stubbing out the parameters, such as controlling the gravity force and the jump frequency.

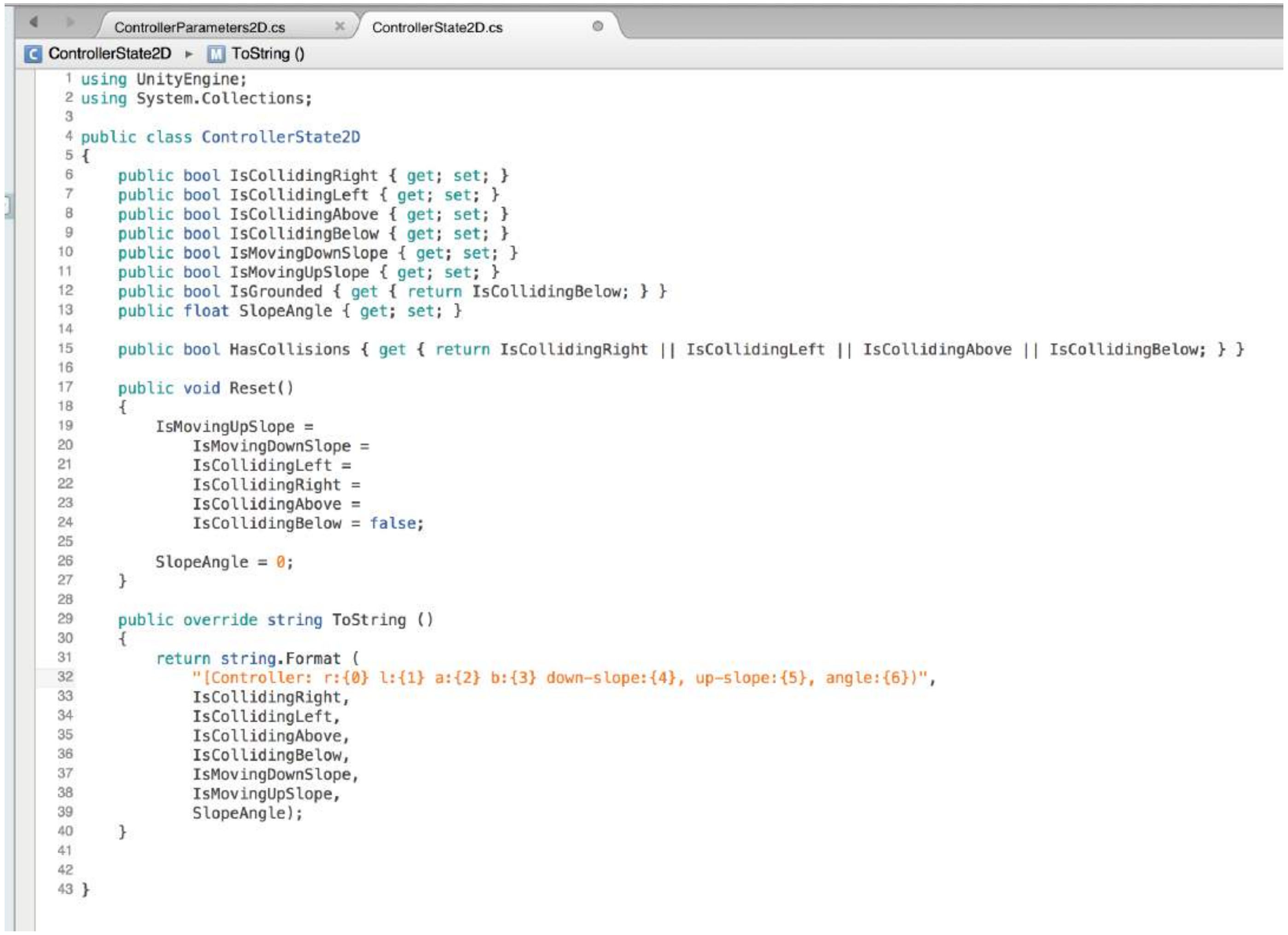

ControllerParameters2D.cs
ControllerState2D.cs
ControllerParameters2D ▸ No selection

```
using UnityEngine;
using System.Collections;

[Serializable]
public class ControllerParameters2D
{
    public enum JumpBehavior
    {
        CanJumpOnGround,
        CanJumpAnywhere,
        CantJump
    }

    public Vector2 MaxVelocity = new Vector2 (float.MaxValue, float.MaxValue);

    [Range(0, 90)]
    public float SlopeLimit = 30;

    public float Gravity = -25f;

    public JumpBehavior JumpRestrictions;

    public float JumpFrequency = .25f;

}
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

I need to now define all of the different states that my player could possibly be in, so are they moving up or down, or are they colliding with a platform.

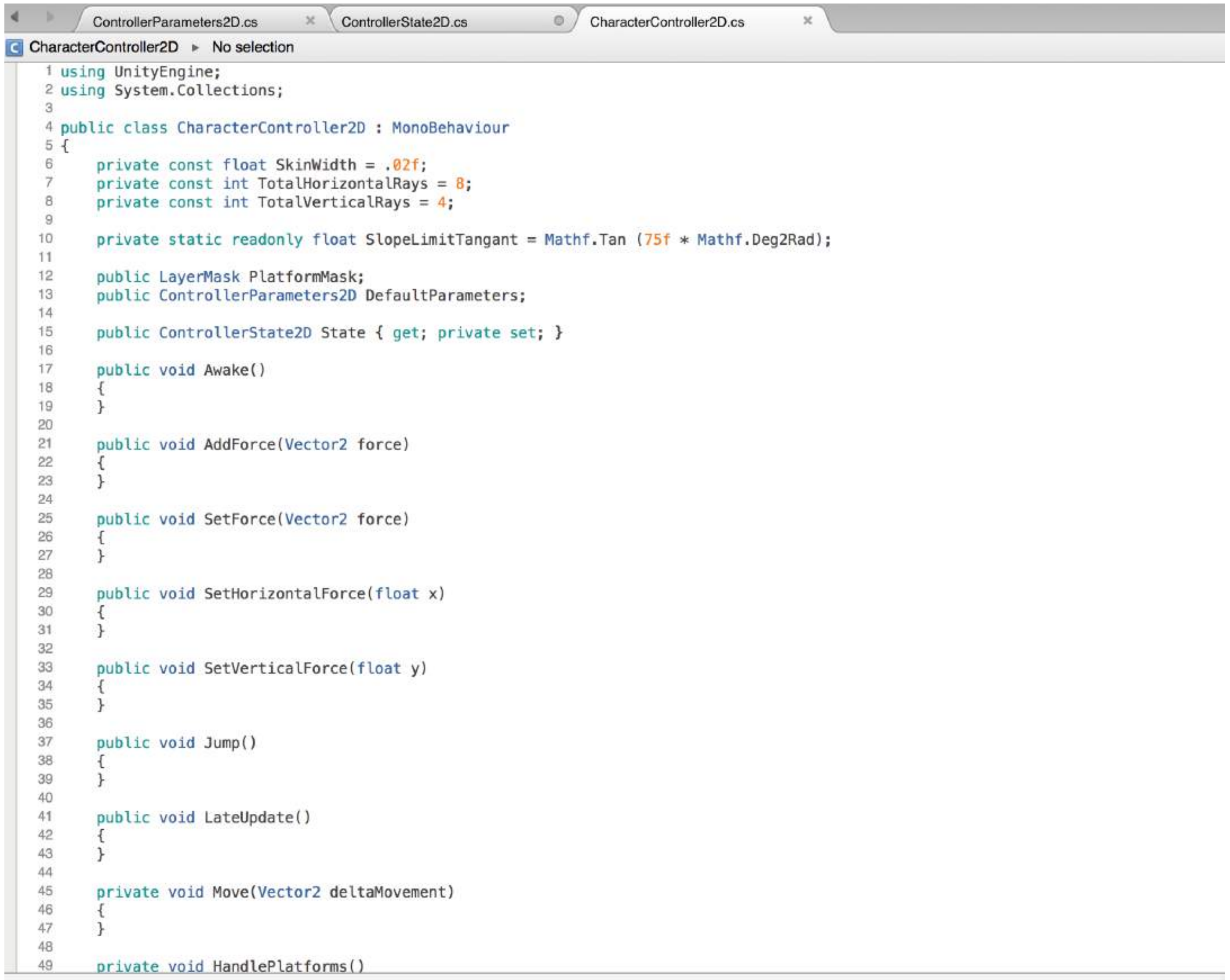


```
using UnityEngine;
using System.Collections;

public class ControllerState2D
{
    public bool IsCollidingRight { get; set; }
    public bool IsCollidingLeft { get; set; }
    public bool IsCollidingAbove { get; set; }
    public bool IsCollidingBelow { get; set; }
    public bool IsMovingDownSlope { get; set; }
    public bool IsMovingUpSlope { get; set; }
    public bool IsGrounded { get { return IsCollidingBelow; } }
    public float SlopeAngle { get; set; }

    public bool HasCollisions { get { return IsCollidingRight || IsCollidingLeft || IsCollidingAbove || IsCollidingBelow; } }

    public void Reset()
    {
        IsMovingUpSlope =
            IsMovingDownSlope =
            IsCollidingLeft =
            IsCollidingRight =
            IsCollidingAbove =
            IsCollidingBelow = false;

        SlopeAngle = 0;
    }

    public override string ToString ()
    {
        return string.Format (
            "[Controller: r:{0} l:{1} a:{2} b:{3} down-slope:{4}, up-slope:{5}, angle:{6})",
            IsCollidingRight,
            IsCollidingLeft,
            IsCollidingAbove,
            IsCollidingBelow,
            IsMovingDownSlope,
            IsMovingUpSlope,
            SlopeAngle);
    }


}
```

- CHARACTER CONTROLLER

The main character controller code (part 1). This is where all of my public feels for the player will be defined.

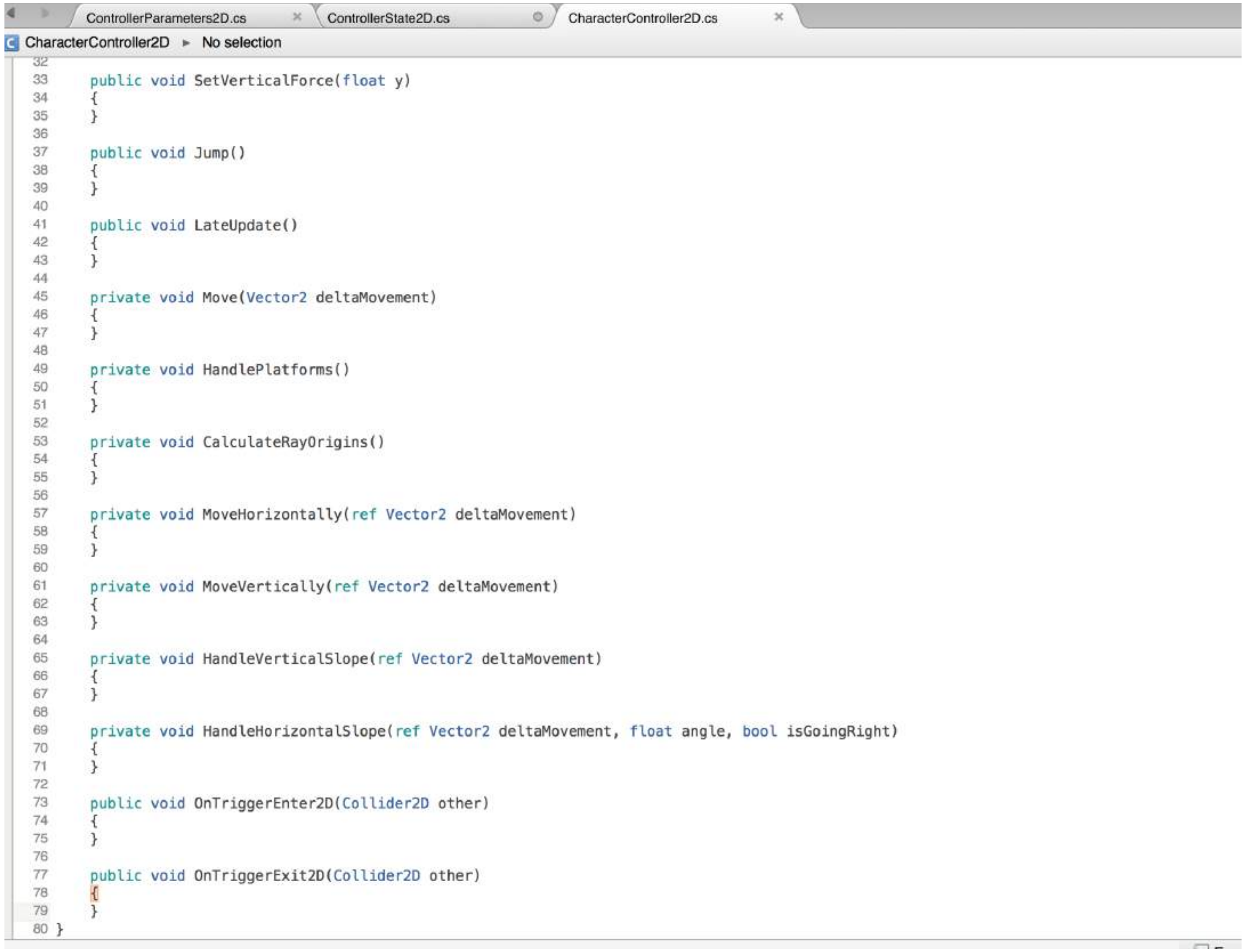
By stubbing out my classes I am able to see what I need to so later on I will be able to create player movement as the player class evokes the character controller 2D class.


ControllerParameters2D.cs
ControllerState2D.cs
CharacterController2D.cs
CharacterController2D ▸ No selection

```
using UnityEngine;
using System.Collections;

public class CharacterController2D : MonoBehaviour
{
    private const float SkinWidth = .02f;
    private const int TotalHorizontalRays = 8;
    private const int TotalVerticalRays = 4;

    private static readonly float SlopeLimitTangant = Mathf.Tan (75f * Mathf.Deg2Rad);

    public LayerMask PlatformMask;
    public ControllerParameters2D DefaultParameters;

    public ControllerState2D State { get; private set; }

    public void Awake()
    {
    }

    public void AddForce(Vector2 force)
    {
    }

    public void SetForce(Vector2 force)
    {
    }

    public void SetHorizontalForce(float x)
    {
    }

    public void SetVerticalForce(float y)
    {
    }

    public void Jump()
    {
    }

    public void LateUpdate()
    {
    }

    private void Move(Vector2 deltaMovement)
    {
    }

    private void HandlePlatforms()
```

- CHARACTER CONTROLLER

Character controller 2D code (part 2).



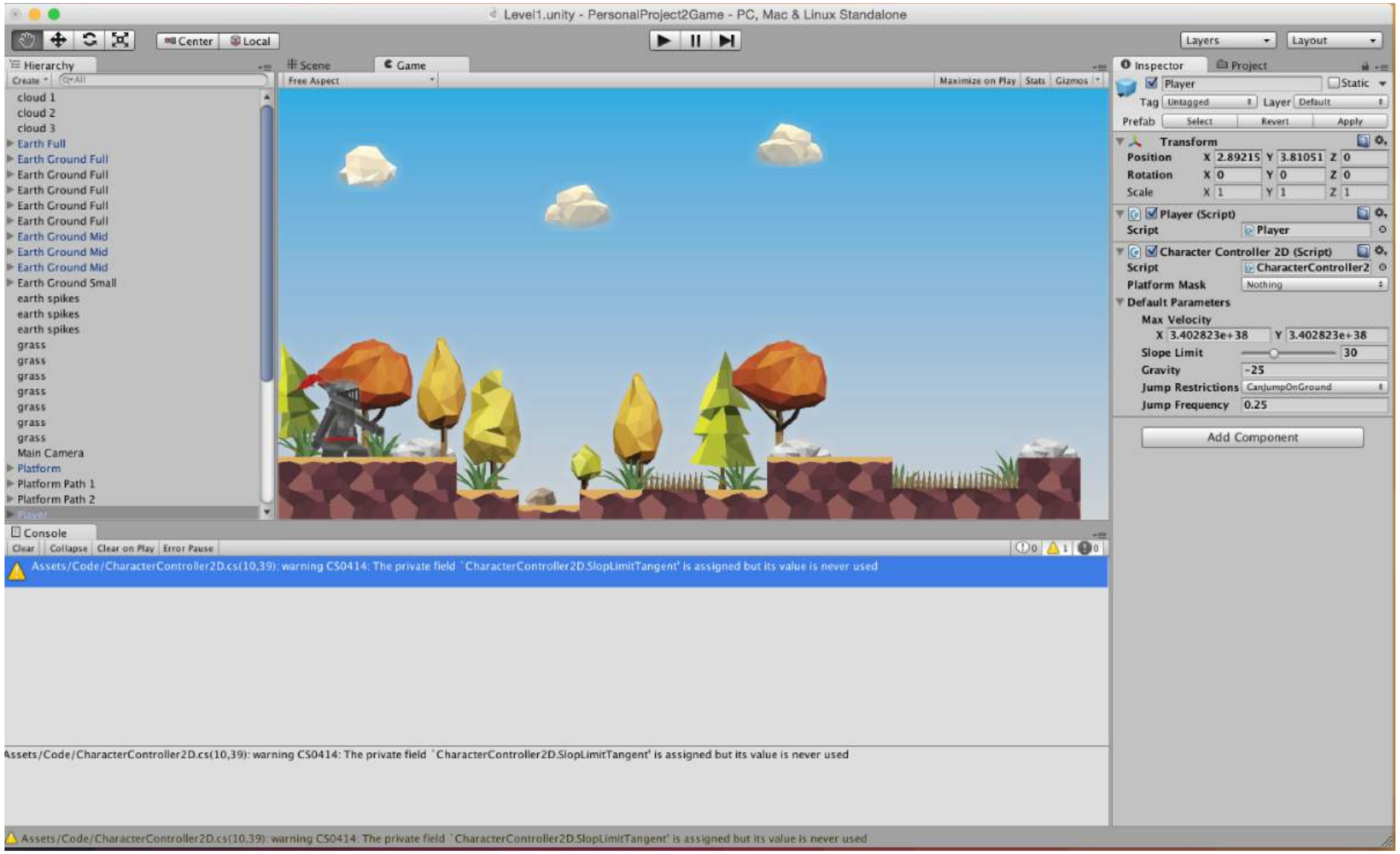
ControllerParameters2D.cs
ControllerState2D.cs
CharacterController2D.cs
CharacterController2D ▸ No selection

```
    public void SetVerticalForce(float y)
    {
    }

    public void Jump()
    {
    }

    public void LateUpdate()
    {
    }

    private void Move(Vector2 deltaMovement)
    {
    }

    private void HandlePlatforms()
    {
    }

    private void CalculateRayOrigins()
    {
    }

    private void MoveHorizontally(ref Vector2 deltaMovement)
    {
    }

    private void MoveVertically(ref Vector2 deltaMovement)
    {
    }

    private void HandleVerticalSlope(ref Vector2 deltaMovement)
    {
    }

    private void HandleHorizontalSlope(ref Vector2 deltaMovement, float angle, bool isGoingRight)
    {
    }

    public void OnTriggerEnter2D(Collider2D other)
    {
    }

    public void OnTriggerExit2D(Collider2D other)
    {
    }
}
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

Back in Unity I need to assign the player script and the character controller 2D script to the actual knight character. By doing this you can see from the code that is written I can easily modify my characters jump and gravity behavior from inside the inspector window.



Player					Static	
Tag	Untagged		Layer	Default		
Prefab	Select		Revert		Apply	
Transform						
Position	X	2.89215	Y	3.81051	Z	0
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1
Player (Script)						
Script	Player					
Character Controller 2D (Script)						
Script	CharacterController2					
Platform Mask	Nothing					
Default Parameters						
Max Velocity	X	3.402823e+38	Y	3.402823e+38		
Slope Limit	30					
Gravity	-25					
Jump Restrictions	CanJumpOnGround					
Jump Frequency	0.25					

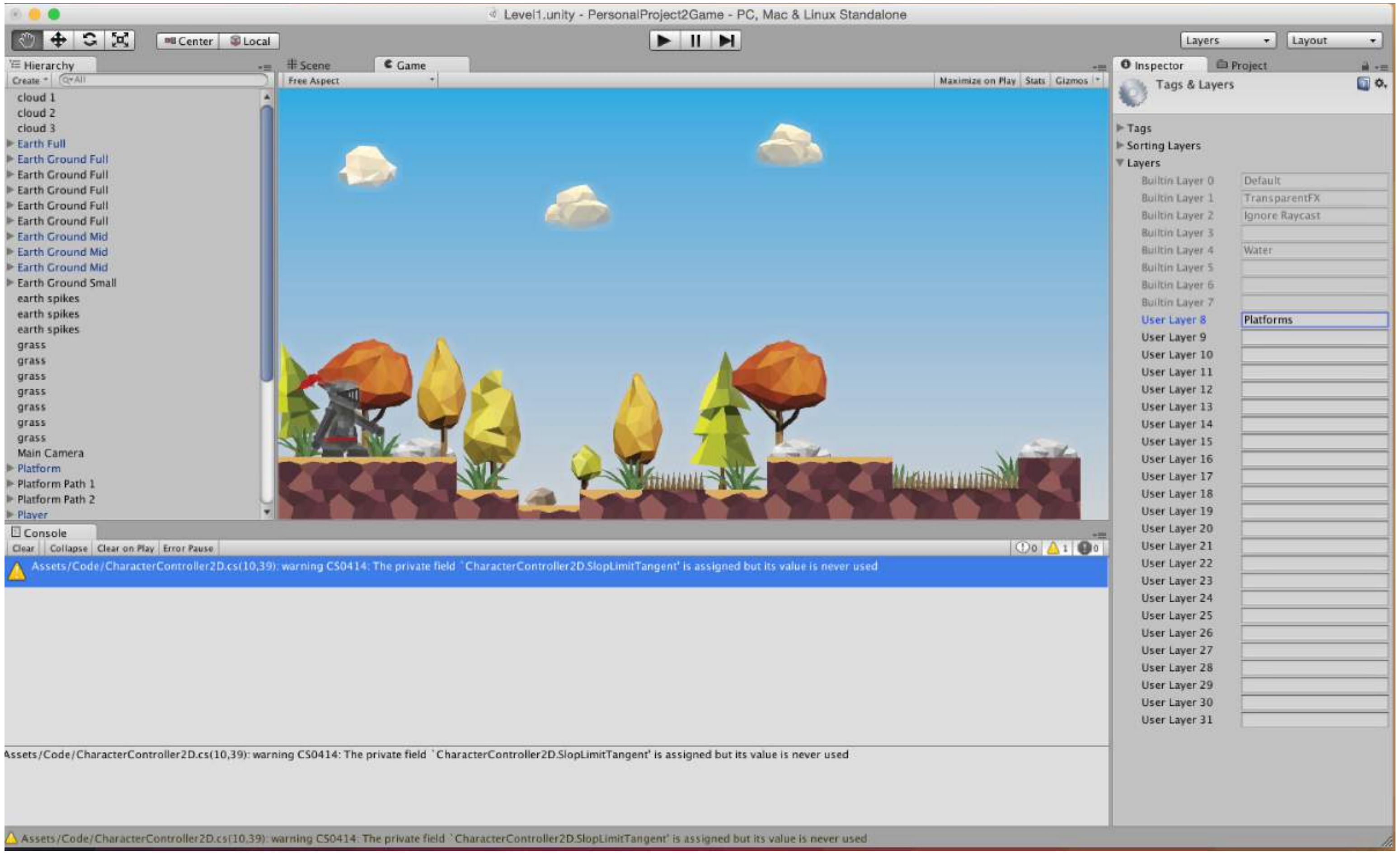
```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used

Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used

Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

I need to put the platforms that the player will run across on a new layer so that everything can be separated out and the player will eventually be on a layer of its own.

In the layers window, I create this new layer and call it 'Platforms'.



Level1.unity - PersonalProject2Game - PC, Mac & Linux Standalone

Hierarchy:
cloud 1
cloud 2
cloud 3
Earth Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Mid
Earth Ground Mid
Earth Ground Mid
Earth Ground Small
earth spikes
earth spikes
earth spikes
grass
grass
grass
grass
grass
grass
grass
Main Camera
Platform
Platform Path 1
Platform Path 2
Player

Inspector – Tags & Layers

Layers

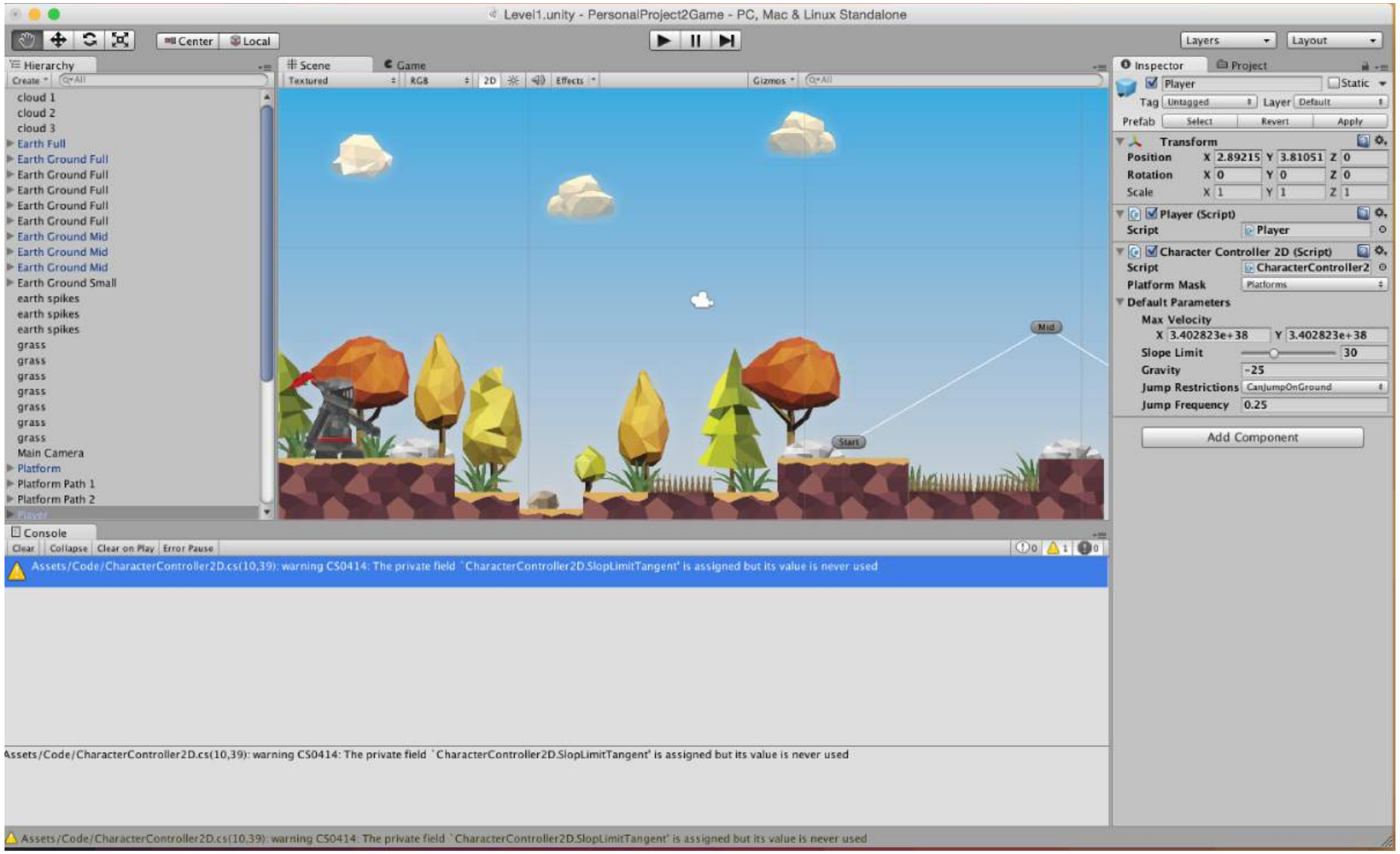
Builtin Layer 0	Default
Builtin Layer 1	TransparentFX
Builtin Layer 2	Ignore Raycast
Builtin Layer 3	
Builtin Layer 4	Water
Builtin Layer 5	
Builtin Layer 6	
Builtin Layer 7	
User Layer 8	Platforms
User Layer 9	
User Layer 10	
User Layer 11	
User Layer 12	
User Layer 13	
User Layer 14	
User Layer 15	
User Layer 16	
User Layer 17	
User Layer 18	
User Layer 19	
User Layer 20	
User Layer 21	
User Layer 22	
User Layer 23	
User Layer 24	
User Layer 25	
User Layer 26	
User Layer 27	
User Layer 28	
User Layer 29	
User Layer 30	
User Layer 31	

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

- CHARACTER CONTROLLER

When I select the player I can now set the platform mask on the player, this will be the layer that the player will follow and stand on.

You can see here I changed my platform mask to the new 'Platforms' layer I just created.



cloud 1
cloud 2
cloud 3
Earth Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Mid
Earth Ground Mid
Earth Ground Mid
Earth Ground Small
earth spikes
earth spikes
earth spikes
grass
grass
grass
grass
grass
grass
grass
Main Camera
Platform
Platform Path 1
Platform Path 2
Player

Player — Static
Tag: Untagged — Layer: Default
Prefab: Select / Revert / Apply

Transform
Position X 2.89215 Y 3.81051 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Player (Script)
Script: Player

Character Controller 2D (Script)
Script: CharacterController2
Platform Mask: Platforms
Default Parameters
Max Velocity X 3.402823e+38 Y 3.402823e+38
Slope Limit 30
Gravity -25
Jump Restrictions: CanJumpOnGround
Jump Frequency 0.25

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

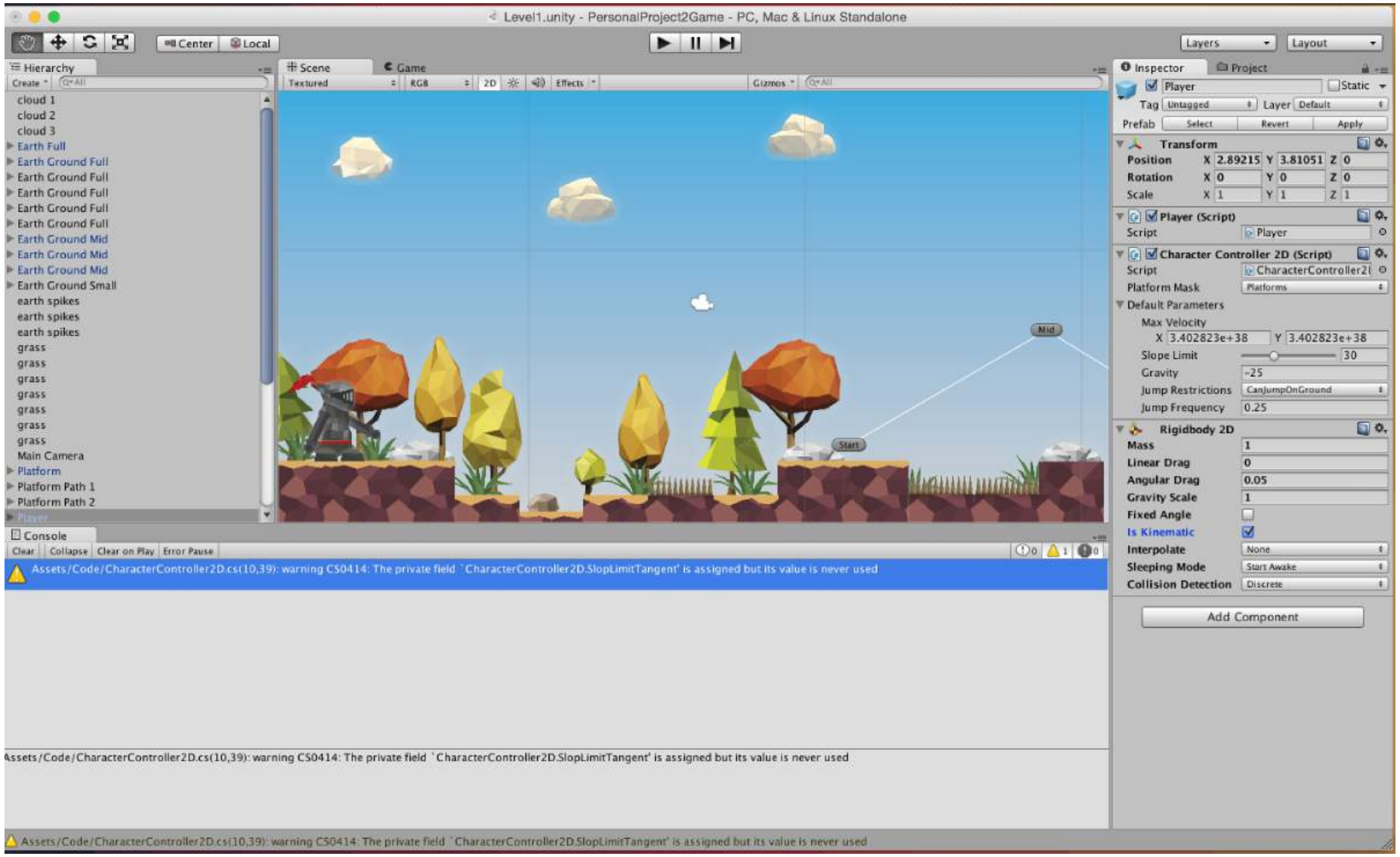
```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

Now I need to add a component called a ‘Ridgidbody 2D’ to the player and select ‘Is Kinematic’.

I do this because as I have written custom physics for my game, making the player ‘Is Kinematic’ will make the player not react is a bad way to gravity and collisions with the objects in the game world so, for example when the player gets hit by an object, by putting a Ridgidbody 2D on it a catapulting effect is created.



Player — Static
Tag: Untagged — Layer: Default
Prefab: Select | Revert | Apply

Transform
Position X 2.89215 Y 3.81051 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Player (Script)
Script: Player

Character Controller 2D (Script)
Script: CharacterController2I
Platform Mask: Platforms
Default Parameters
Max Velocity X 3.402823e+38 Y 3.402823e+38
Slope Limit: 30
Gravity: -25
Jump Restrictions: CanJumpOnGround
Jump Frequency: 0.25

Rigidbody 2D
Mass: 1
Linear Drag: 0
Angular Drag: 0.05
Gravity Scale: 1
Fixed Angle: ☐
Is Kinematic: ✓
Interpolate: None
Sleeping Mode: Start Awake
Collision Detection: Discrete

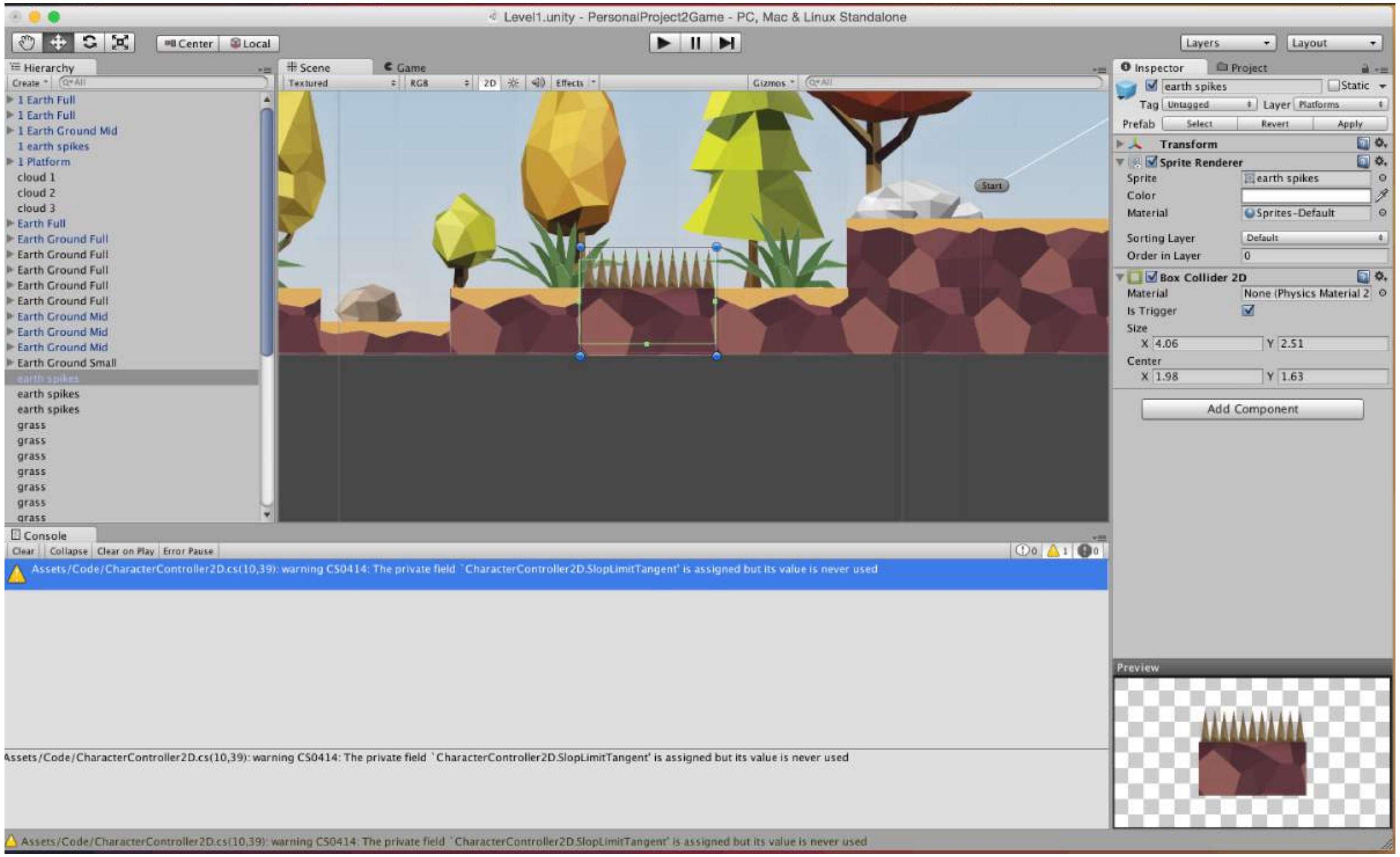
```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

- CHARACTER CONTROLLER

I sort the platforms on the 'Platforms' layer and add a box collider to them, this box collider is what the player will interact with in the world to determine whether to stand on it as at the moment the platforms are just graphics, they don't actually do anything.



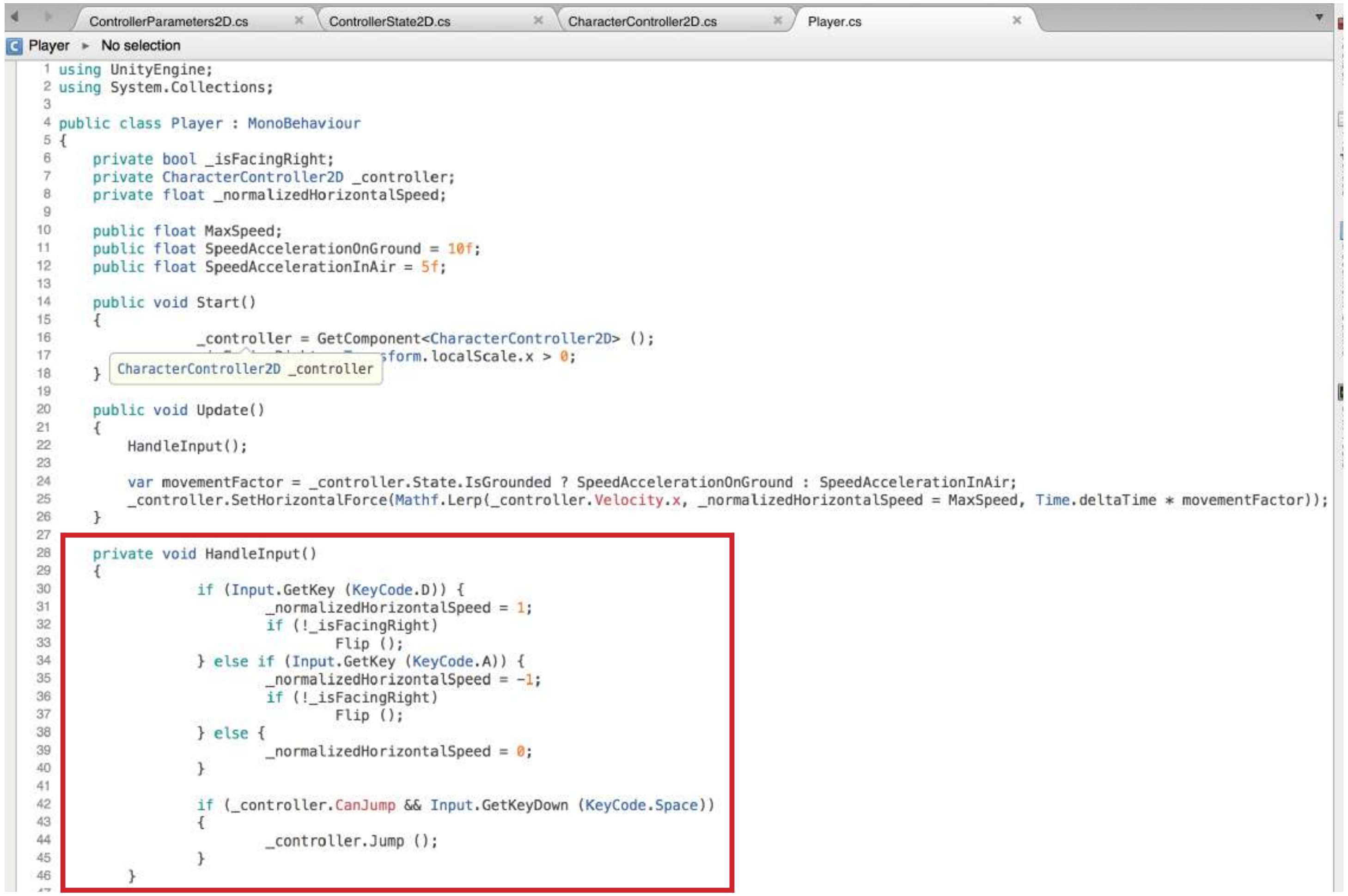
```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

```
Assets/Code/CharacterController2D.cs(10,39): warning CS0414: The private field `CharacterController2D.SlopLimitTangent' is assigned but its value is never used
```

- CHARACTER CONTROLLER

In my Player script I need to sort my controller. So the keyboard inputs need to be defined for the moment. (Part 1).

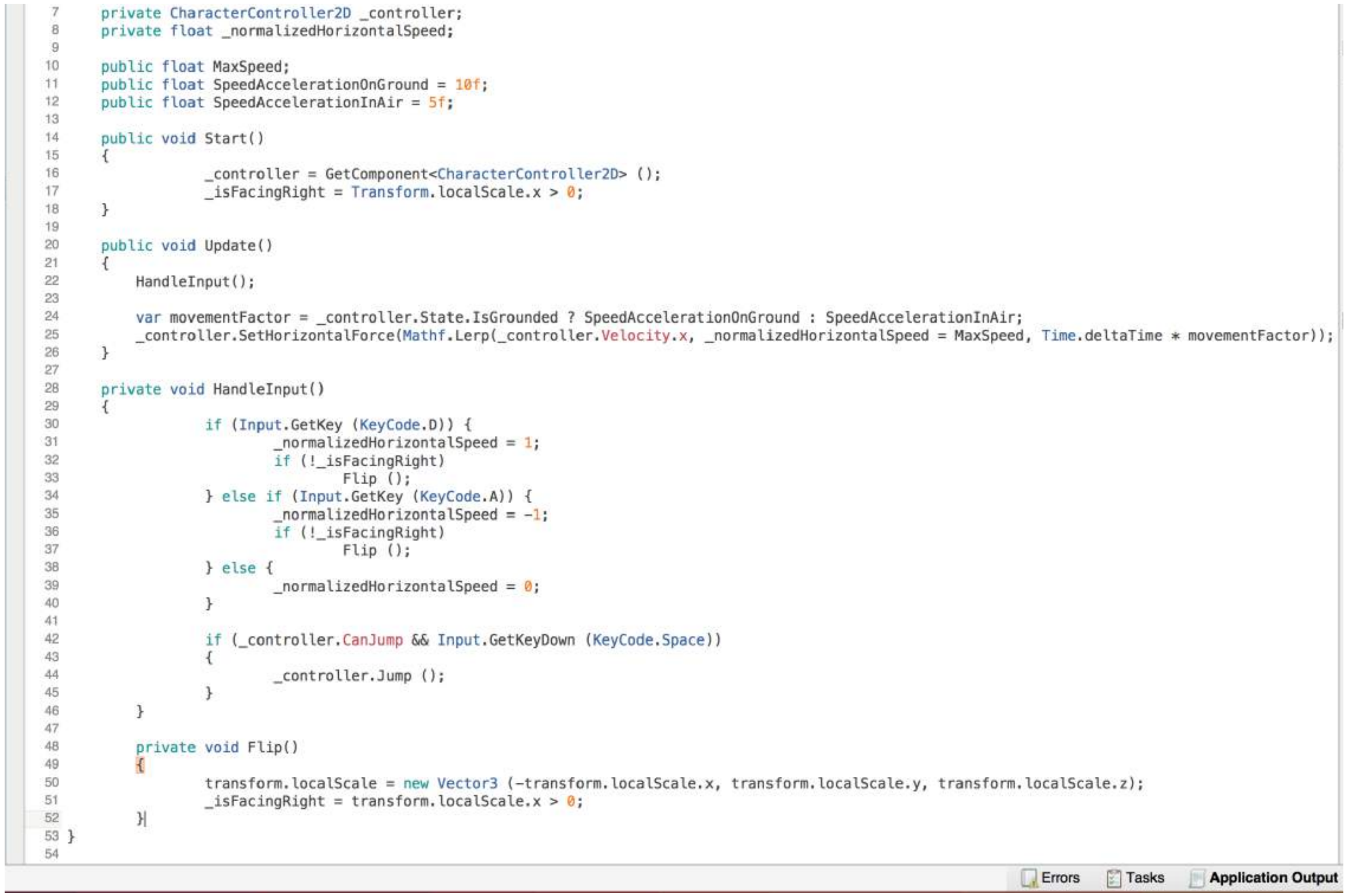


```
using UnityEngine;
using System.Collections;

public class Player : MonoBehaviour
{
    private bool _isFacingRight;
    private CharacterController2D _controller;
    private float _normalizedHorizontalSpeed;

    public float MaxSpeed;
    public float SpeedAccelerationOnGround = 10f;
    public float SpeedAccelerationInAir = 5f;

    public void Start()
    {
            _controller = GetComponent<CharacterController2D> ();
            [illegible]sform.localScale.x > 0;
    }

    public void Update()
    {
        HandleInput();

        var movementFactor = _controller.State.IsGrounded ? SpeedAccelerationOnGround : SpeedAccelerationInAir;
        _controller.SetHorizontalForce(Mathf.Lerp(_controller.Velocity.x, _normalizedHorizontalSpeed = MaxSpeed, Time.deltaTime * movementFactor));
    }

    private void HandleInput()
    {
            if (Input.GetKey (KeyCode.D)) {
                    _normalizedHorizontalSpeed = 1;
                    if (!_isFacingRight)
                            Flip ();
            } else if (Input.GetKey (KeyCode.A)) {
                    _normalizedHorizontalSpeed = -1;
                    if (!_isFacingRight)
                            Flip ();
            } else {
                    _normalizedHorizontalSpeed = 0;
            }

            if (_controller.CanJump && Input.GetKeyDown (KeyCode.Space))
            {
                    _controller.Jump ();
            }
        }
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

(Part 2) is showing the 'flip' method, so the player will flip left or right depending on which input has been pressed by the user.



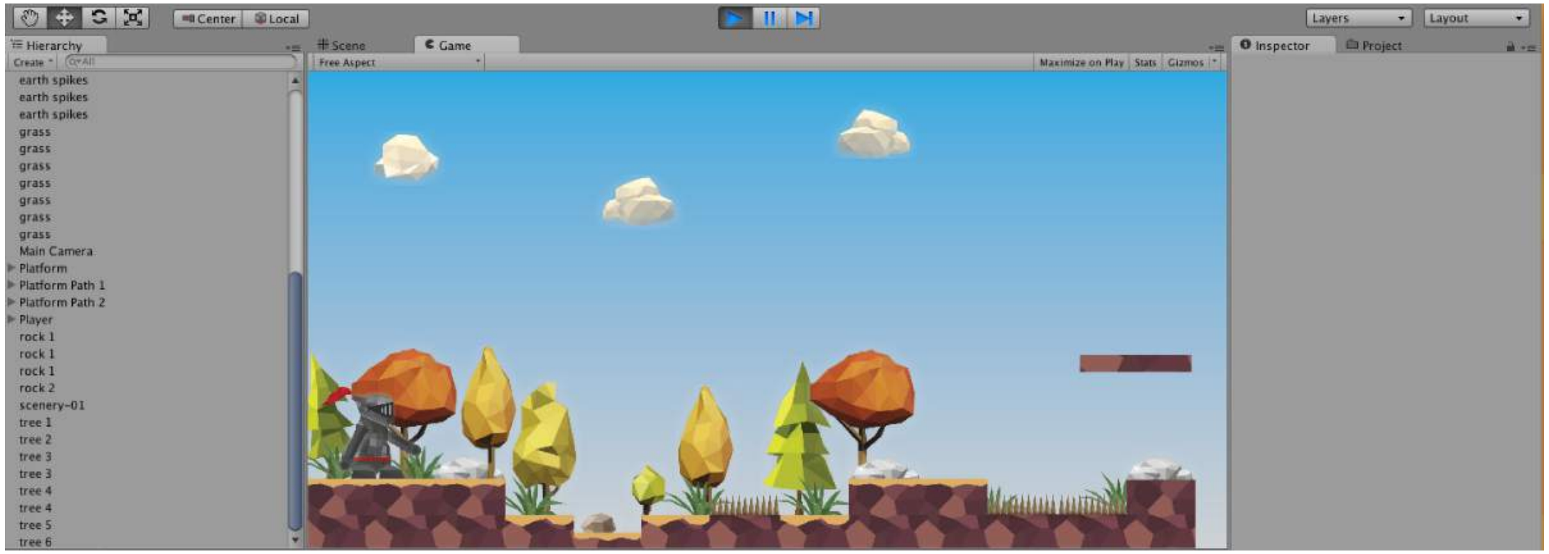
```
    private CharacterController2D _controller;
    private float _normalizedHorizontalSpeed;

    public float MaxSpeed;
    public float SpeedAccelerationOnGround = 10f;
    public float SpeedAccelerationInAir = 5f;

    public void Start()
    {
                _controller = GetComponent<CharacterController2D> ();
                _isFacingRight = Transform.localScale.x > 0;
    }

    public void Update()
    {
        HandleInput();

        var movementFactor = _controller.State.IsGrounded ? SpeedAccelerationOnGround : SpeedAccelerationInAir;
        _controller.SetHorizontalForce(Mathf.Lerp(_controller.Velocity.x, _normalizedHorizontalSpeed = MaxSpeed, Time.deltaTime * movementFactor));
    }

    private void HandleInput()
    {
                if (Input.GetKey (KeyCode.D)) {
                        _normalizedHorizontalSpeed = 1;
                        if (!_isFacingRight)
                                Flip ();
                } else if (Input.GetKey (KeyCode.A)) {
                        _normalizedHorizontalSpeed = -1;
                        if (!_isFacingRight)
                                Flip ();
                } else {
                        _normalizedHorizontalSpeed = 0;
                }

                if (_controller.CanJump && Input.GetKeyDown (KeyCode.Space))
                {
                        _controller.Jump ();
                }
        }

        private void Flip()
        {
                transform.localScale = new Vector3 (-transform.localScale.x, transform.localScale.y, transform.localScale.z);
                _isFacingRight = transform.localScale.x > 0;
        }
}
```

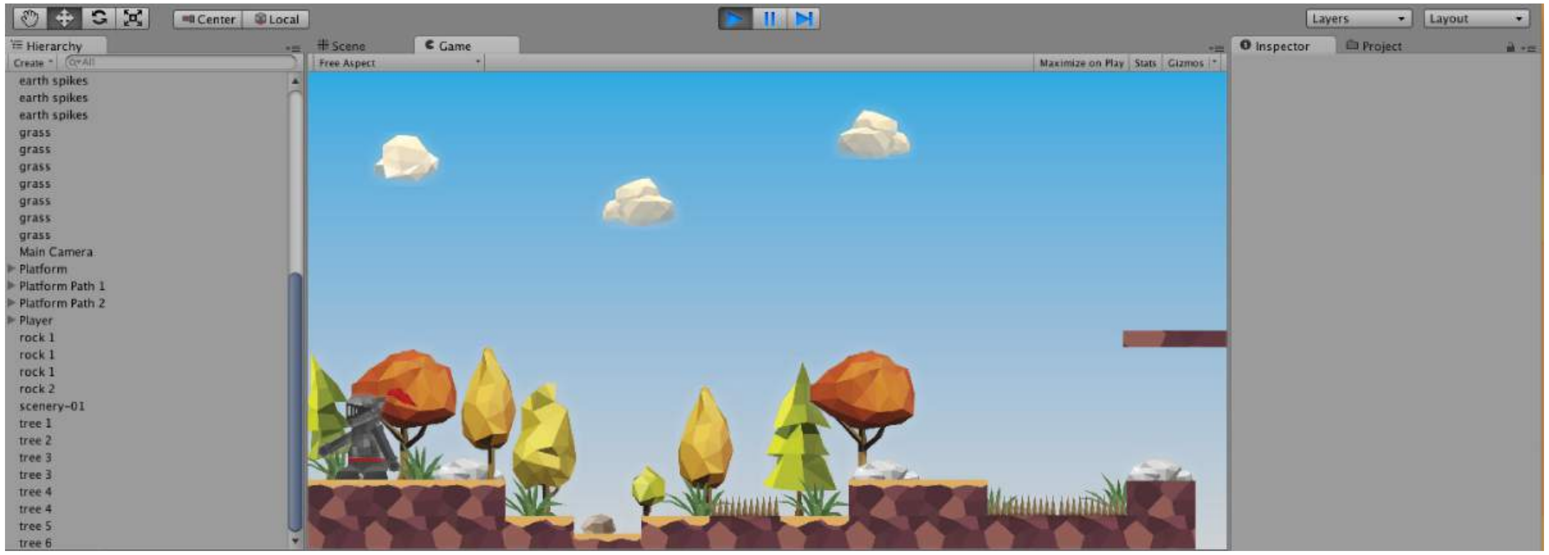
GAME DEVELOPMENT

- CHARACTER CONTROLLER

Back in Unity, when I press play and press the 'D' key on my laptop the player faces right and when I press the 'A' key the player faces left.

Now that I have defined these inputs in Unity, when it comes to build the glove controller, all I will need to do is change the input from the 'D' key etc. to whatever the numerical value is from Arduino and hopefully it will work.

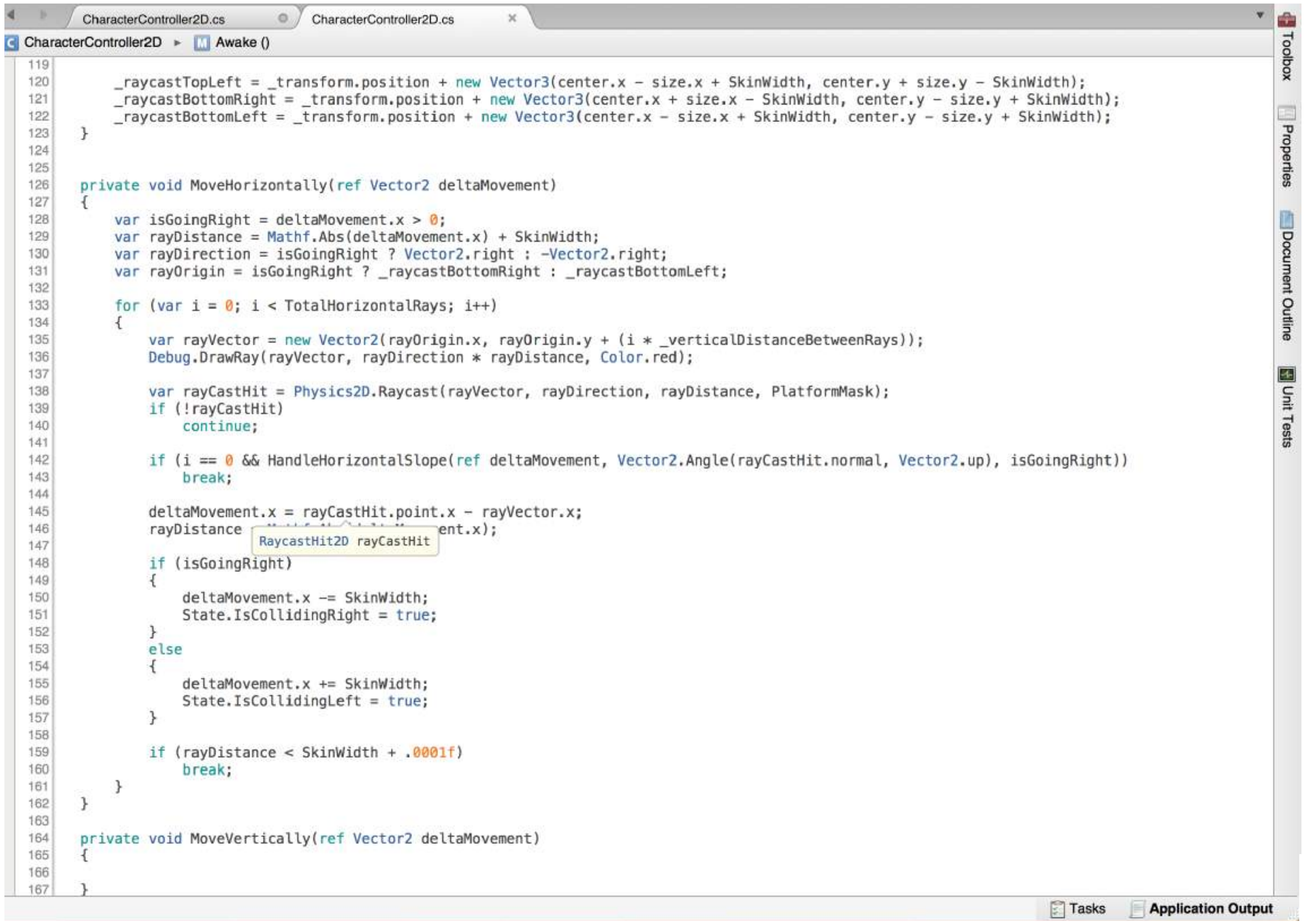




- CHARACTER CONTROLLER

Now that I have got the player facing left and right, I need to now make it move across the world. I will work on making the player move horizontally first.

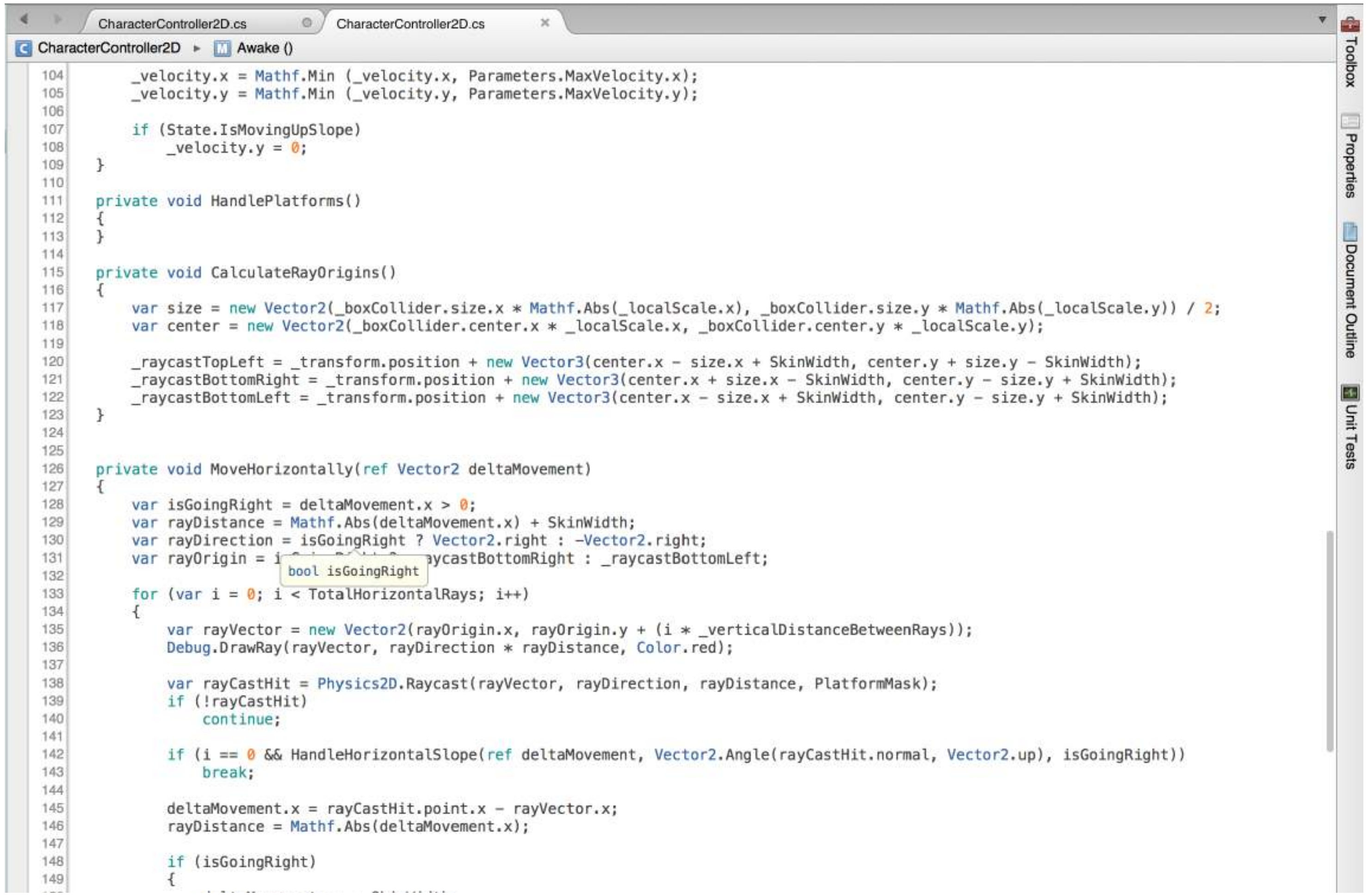
In the move horizontally class in the CharacterController2D script I add code to allow me to do this. (Part 1)



```
        _raycastTopLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y + size.y - SkinWidth);
        _raycastBottomRight = _transform.position + new Vector3(center.x + size.x - SkinWidth, center.y - size.y + SkinWidth);
        _raycastBottomLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y - size.y + SkinWidth);
    }


    private void MoveHorizontally(ref Vector2 deltaMovement)
    {
        var isGoingRight = deltaMovement.x > 0;
        var rayDistance = Mathf.Abs(deltaMovement.x) + SkinWidth;
        var rayDirection = isGoingRight ? Vector2.right : -Vector2.right;
        var rayOrigin = isGoingRight ? _raycastBottomRight : _raycastBottomLeft;

        for (var i = 0; i < TotalHorizontalRays; i++)
        {
            var rayVector = new Vector2(rayOrigin.x, rayOrigin.y + (i * _verticalDistanceBetweenRays));
            Debug.DrawRay(rayVector, rayDirection * rayDistance, Color.red);

            var rayCastHit = Physics2D.Raycast(rayVector, rayDirection, rayDistance, PlatformMask);
            if (!rayCastHit)
                continue;

            if (i == 0 && HandleHorizontalSlope(ref deltaMovement, Vector2.Angle(rayCastHit.normal, Vector2.up), isGoingRight))
                break;

            deltaMovement.x = rayCastHit.point.x - rayVector.x;
            rayDistance [illegible]ent.x);

            if (isGoingRight)
            {
                deltaMovement.x -= SkinWidth;
                State.IsCollidingRight = true;
            }
            else
            {
                deltaMovement.x += SkinWidth;
                State.IsCollidingLeft = true;
            }

            if (rayDistance < SkinWidth + .0001f)
                break;
        }
    }

    private void MoveVertically(ref Vector2 deltaMovement)
    {

    }
```

- CHARACTER CONTROLLER

A continuation (Part 2) of the CharacterController2D script to make the player move around the world.

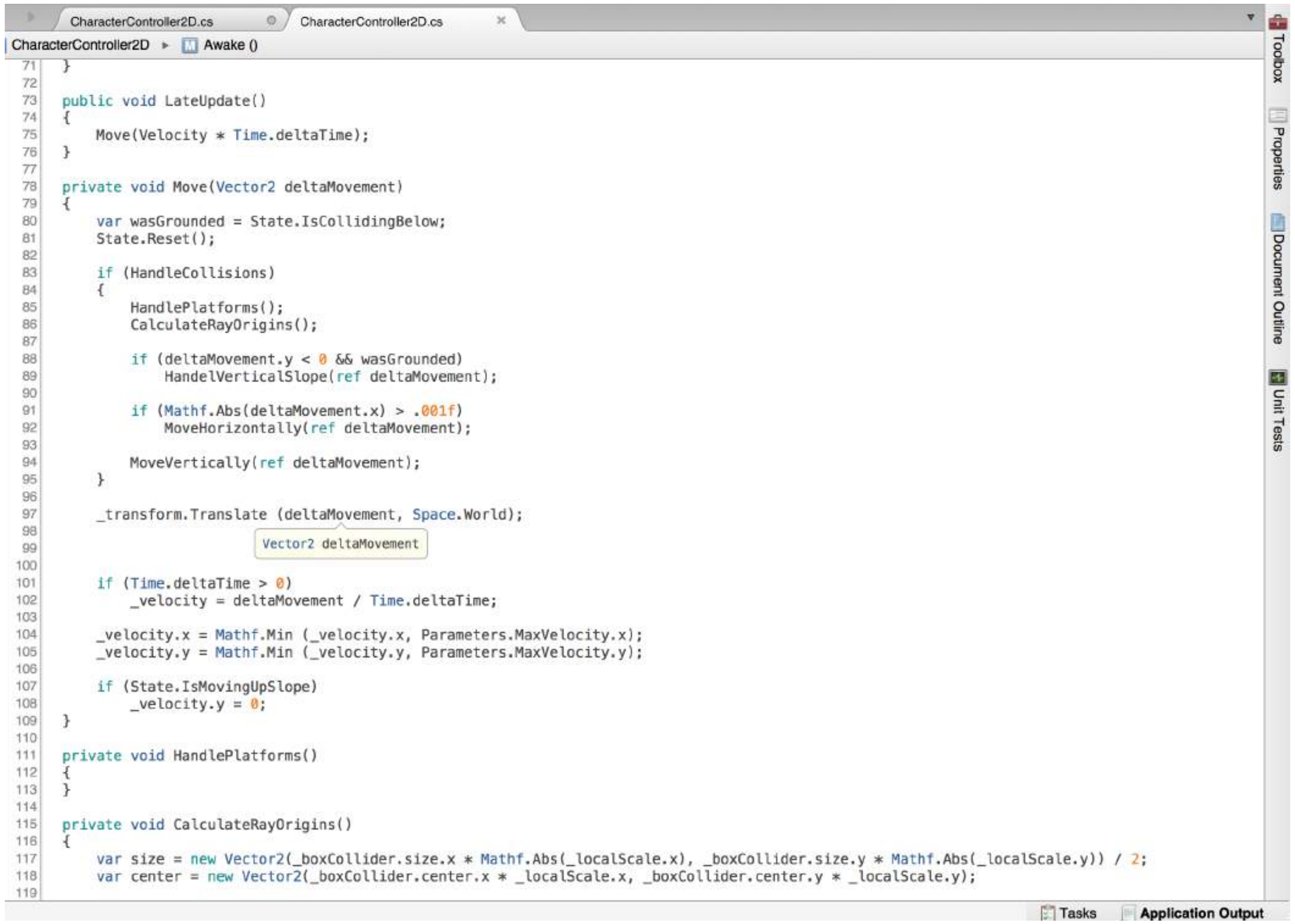


```
        _velocity.x = Mathf.Min (_velocity.x, Parameters.MaxVelocity.x);
        _velocity.y = Mathf.Min (_velocity.y, Parameters.MaxVelocity.y);

        if (State.IsMovingUpSlope)
            _velocity.y = 0;
    }

    private void HandlePlatforms()
    {
    }

    private void CalculateRayOrigins()
    {
        var size = new Vector2(_boxCollider.size.x * Mathf.Abs(_localScale.x), _boxCollider.size.y * Mathf.Abs(_localScale.y)) / 2;
        var center = new Vector2(_boxCollider.center.x * _localScale.x, _boxCollider.center.y * _localScale.y);

        _raycastTopLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y + size.y - SkinWidth);
        _raycastBottomRight = _transform.position + new Vector3(center.x + size.x - SkinWidth, center.y - size.y + SkinWidth);
        _raycastBottomLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y - size.y + SkinWidth);
    }


    private void MoveHorizontally(ref Vector2 deltaMovement)
    {
        var isGoingRight = deltaMovement.x > 0;
        var rayDistance = Mathf.Abs(deltaMovement.x) + SkinWidth;
        var rayDirection = isGoingRight ? Vector2.right : -Vector2.right;
        var rayOrigin = i                   aycastBottomRight : _raycastBottomLeft;

        for (var i = 0; i < TotalHorizontalRays; i++)
        {
            var rayVector = new Vector2(rayOrigin.x, rayOrigin.y + (i * _verticalDistanceBetweenRays));
            Debug.DrawRay(rayVector, rayDirection * rayDistance, Color.red);

            var rayCastHit = Physics2D.Raycast(rayVector, rayDirection, rayDistance, PlatformMask);
            if (!rayCastHit)
                continue;

            if (i == 0 && HandleHorizontalSlope(ref deltaMovement, Vector2.Angle(rayCastHit.normal, Vector2.up), isGoingRight))
                break;

            deltaMovement.x = rayCastHit.point.x - rayVector.x;
            rayDistance = Mathf.Abs(deltaMovement.x);

            if (isGoingRight)
            {
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

Part 3 of the code.

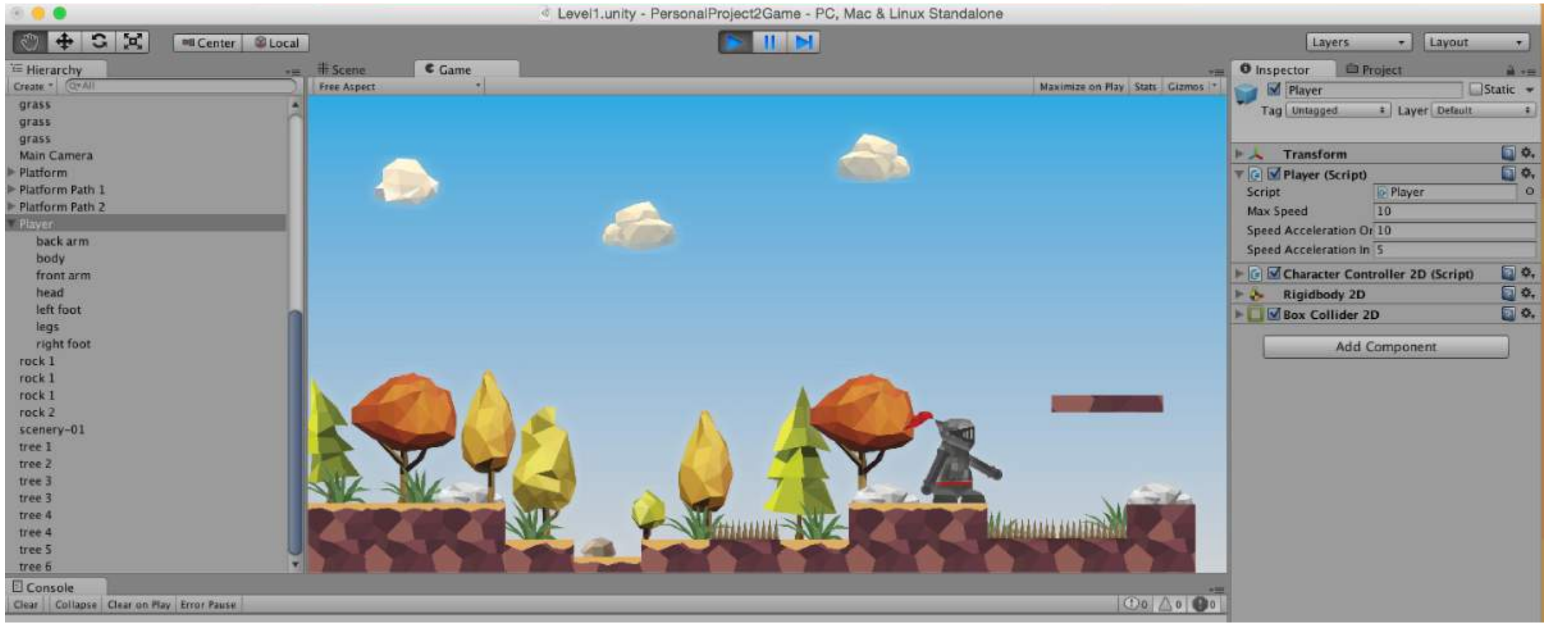


```
}

public void LateUpdate()
{
    Move(Velocity * Time.deltaTime);
}

private void Move(Vector2 deltaMovement)
{
    var wasGrounded = State.IsCollidingBelow;
    State.Reset();

    if (HandleCollisions)
    {
        HandlePlatforms();
        CalculateRayOrigins();

        if (deltaMovement.y < 0 && wasGrounded)
            HandelVerticalSlope(ref deltaMovement);

        if (Mathf.Abs(deltaMovement.x) > .001f)
            MoveHorizontally(ref deltaMovement);

        MoveVertically(ref deltaMovement);
    }

    _transform.Translate (deltaMovement, Space.World);


    if (Time.deltaTime > 0)
        _velocity = deltaMovement / Time.deltaTime;

    _velocity.x = Mathf.Min (_velocity.x, Parameters.MaxVelocity.x);
    _velocity.y = Mathf.Min (_velocity.y, Parameters.MaxVelocity.y);

    if (State.IsMovingUpSlope)
        _velocity.y = 0;
}

private void HandlePlatforms()
{
}

private void CalculateRayOrigins()
{
    var size = new Vector2(_boxCollider.size.x * Mathf.Abs(_localScale.x), _boxCollider.size.y * Mathf.Abs(_localScale.y)) / 2;
    var center = new Vector2(_boxCollider.center.x * _localScale.x, _boxCollider.center.y * _localScale.y);

```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

When I hit play back in Unity I am able to move left and right around the world, which is what I want the inputs to do.



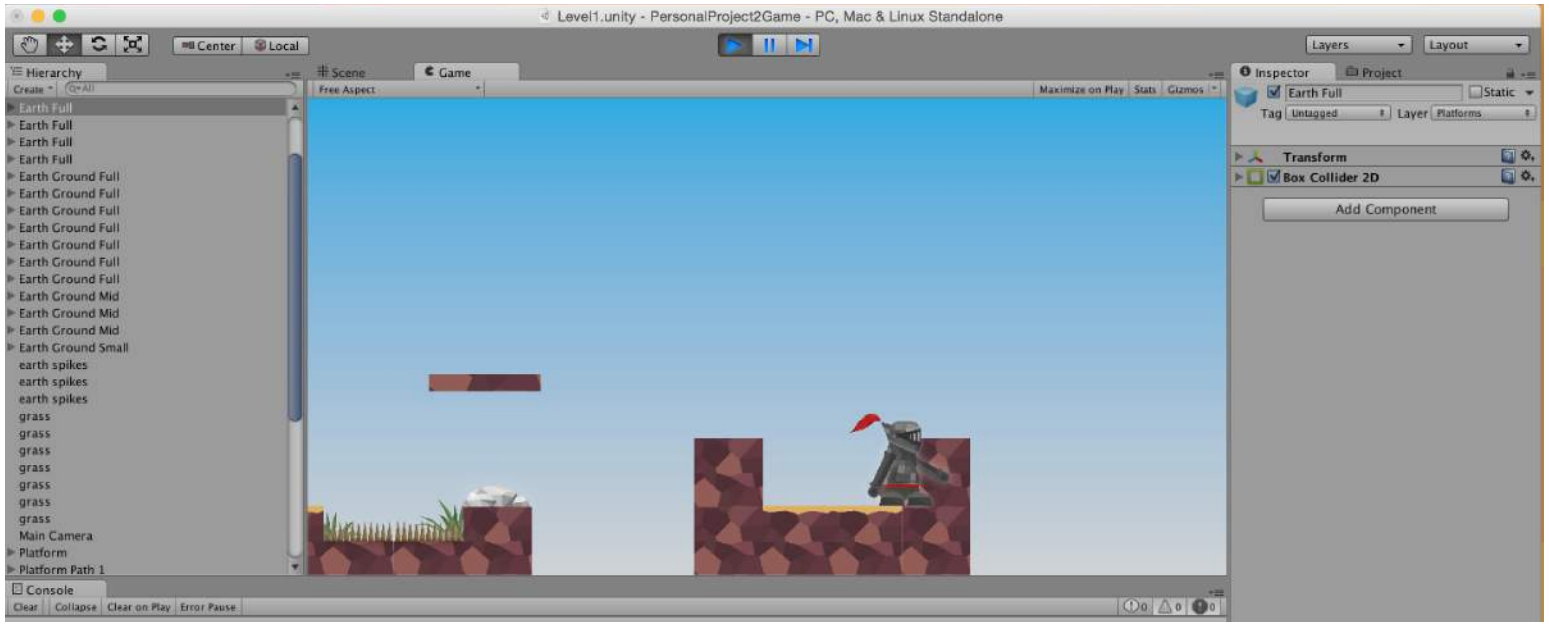
Player
Static
Tag Untagged
Layer Default

Transform
Player (Script)
Script Player
Max Speed 10
Speed Acceleration On 10
Speed Acceleration In 5
Character Controller 2D (Script)
Rigidbody 2D
Box Collider 2D

GAME DEVELOPMENT

- CHARACTER CONTROLLER

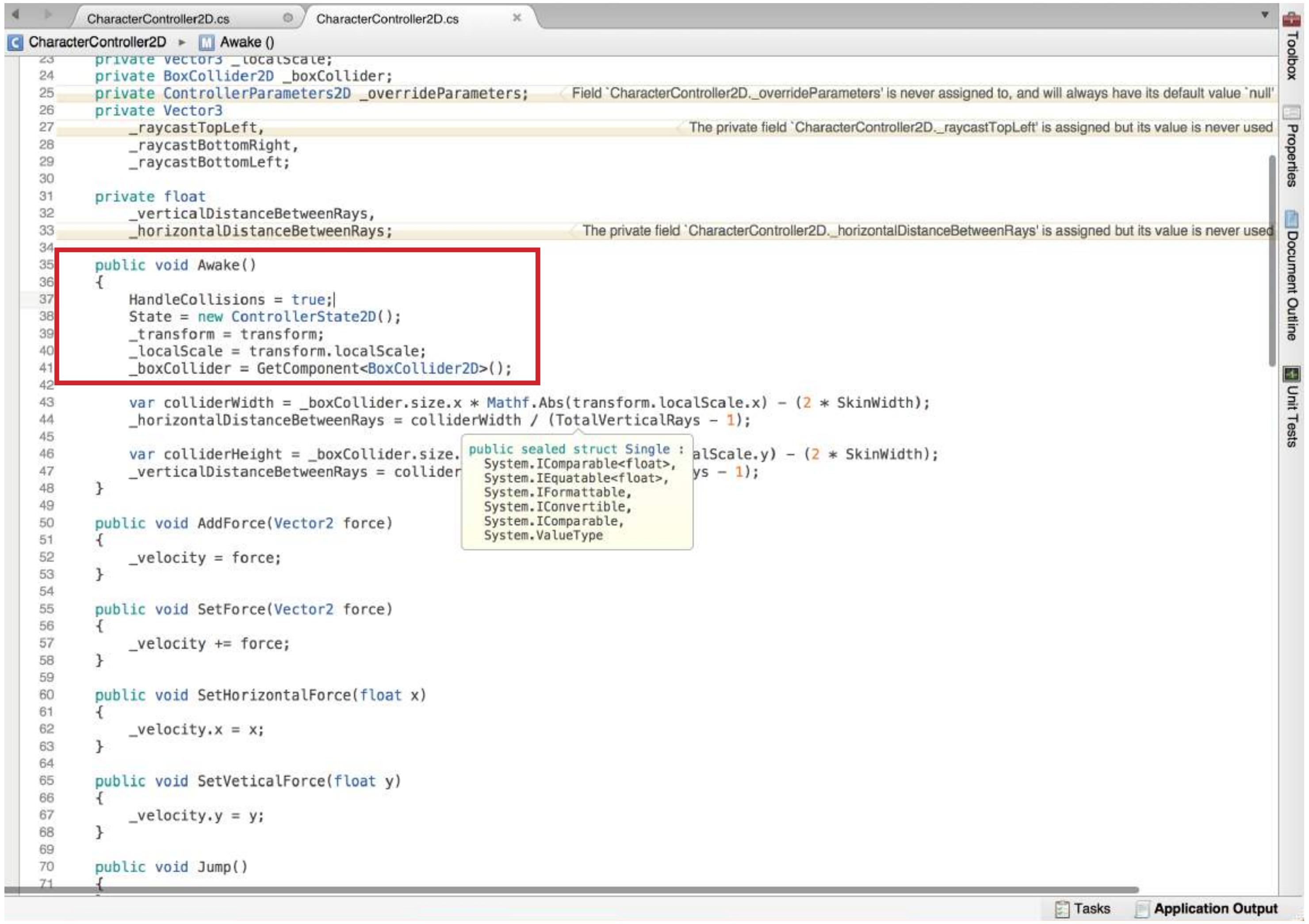
While experimenting with the platforms, I noticed that my handle collisions between the player and the platforms are not working right, the player goes half way through the platform! I will need to get this issue fixed as soon as possible.



- CHARACTER CONTROLLER

The error was actually really simple to fix. Back in the CharacterController2D I was greeted by a couple of errors anyway that pointed me to the problem.

All I needed to do was in the 'public void Awake()' method make handle collisions to 'true' and this should be the fix.



```
    private Vector3 _localScale;
    private BoxCollider2D _boxCollider;
    private ControllerParameters2D _overrideParameters;
    private Vector3
        _raycastTopLeft,
        _raycastBottomRight,
        _raycastBottomLeft;

    private float
        _verticalDistanceBetweenRays,
        _horizontalDistanceBetweenRays;

    public void Awake()
    {
        HandleCollisions = true;
        State = new ControllerState2D();
        _transform = transform;
        _localScale = transform.localScale;
        _boxCollider = GetComponent<BoxCollider2D>();

        var colliderWidth = _boxCollider.size.x * Mathf.Abs(transform.localScale.x) - (2 * SkinWidth);
        _horizontalDistanceBetweenRays = colliderWidth / (TotalVerticalRays - 1);

        var colliderHeight = _boxCollider.size.                          alScale.y) - (2 * SkinWidth);
        _verticalDistanceBetweenRays = collider                          ys - 1);
    }

    public void AddForce(Vector2 force)
    {
        _velocity = force;
    }

    public void SetForce(Vector2 force)
    {
        _velocity += force;
    }

    public void SetHorizontalForce(float x)
    {
        _velocity.x = x;
    }

    public void SetVeticalForce(float y)
    {
        _velocity.y = y;
    }

    public void Jump()
    {
```

Field `CharacterController2D._overrideParameters' is never assigned to, and will always have its default value `null'

The private field `CharacterController2D._raycastTopLeft' is assigned but its value is never used

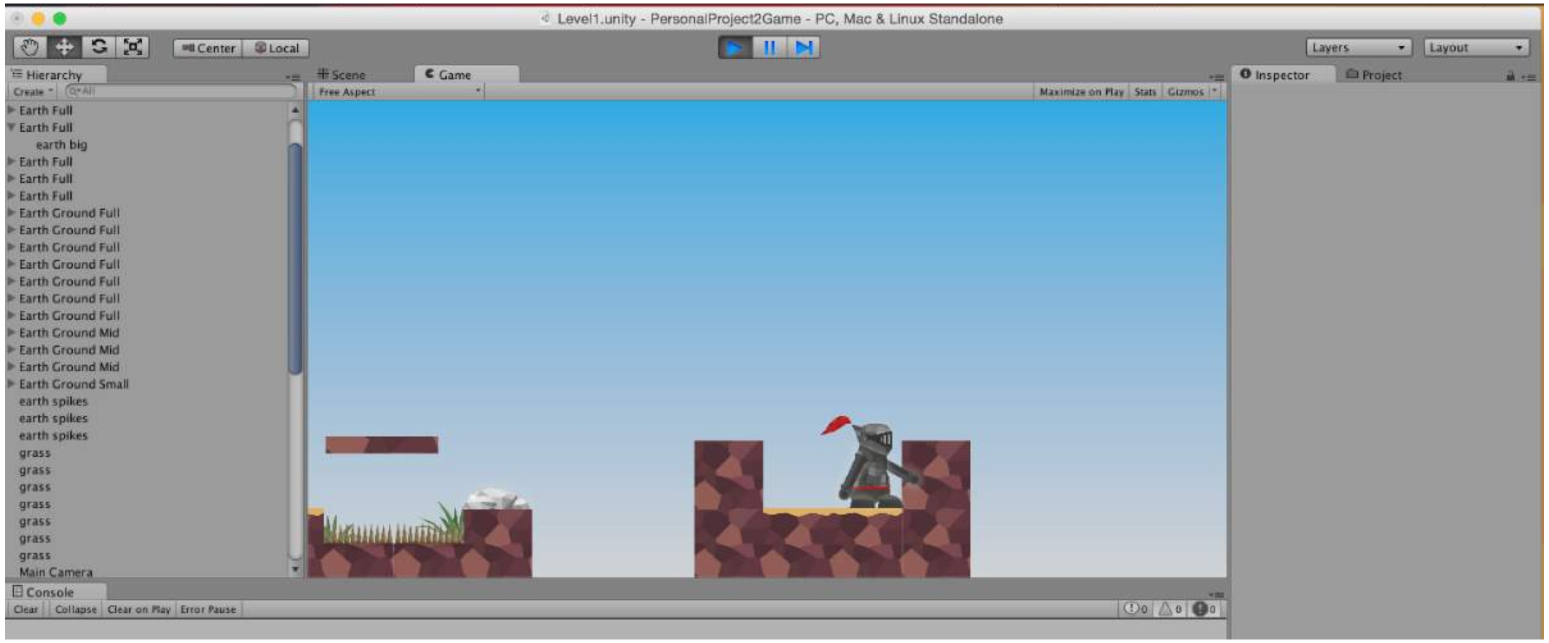
The private field `CharacterController2D._horizontalDistanceBetweenRays' is assigned but its value is never used

```
public sealed struct Single :
  System.IComparable<float>,
  System.IEquatable<float>,
  System.IFormattable,
  System.IConvertible,
  System.IComparable,
  System.ValueType
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

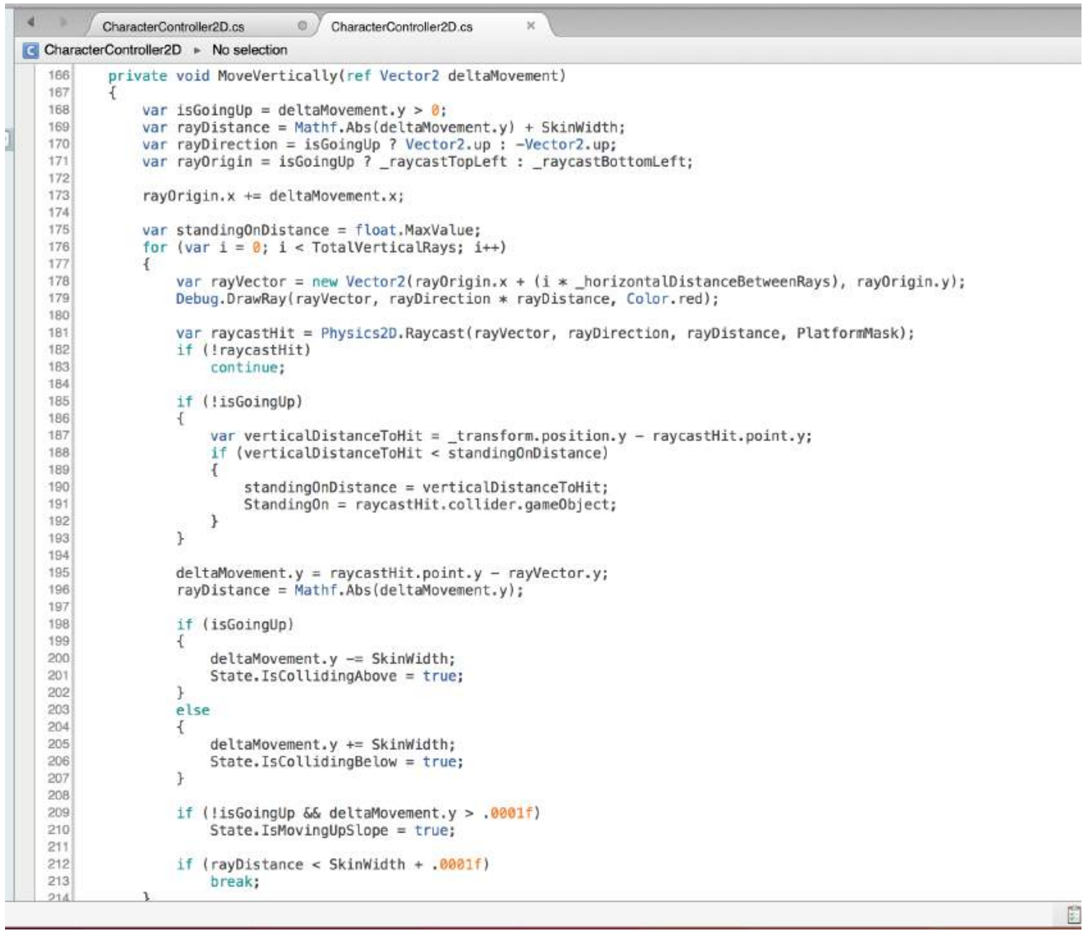
Testing the game back in Unity in the same position, as before, when I move towards the raised platform, the player collides with it and can go no further past it that means that the fix I just did is working!



GAME DEVELOPMENT

- CHARACTER CONTROLLER

I now need to make the character able to move vertically. In the 'move vertically' class I add similar code the horizontal class just change some of it so the movement is on the Y axis.

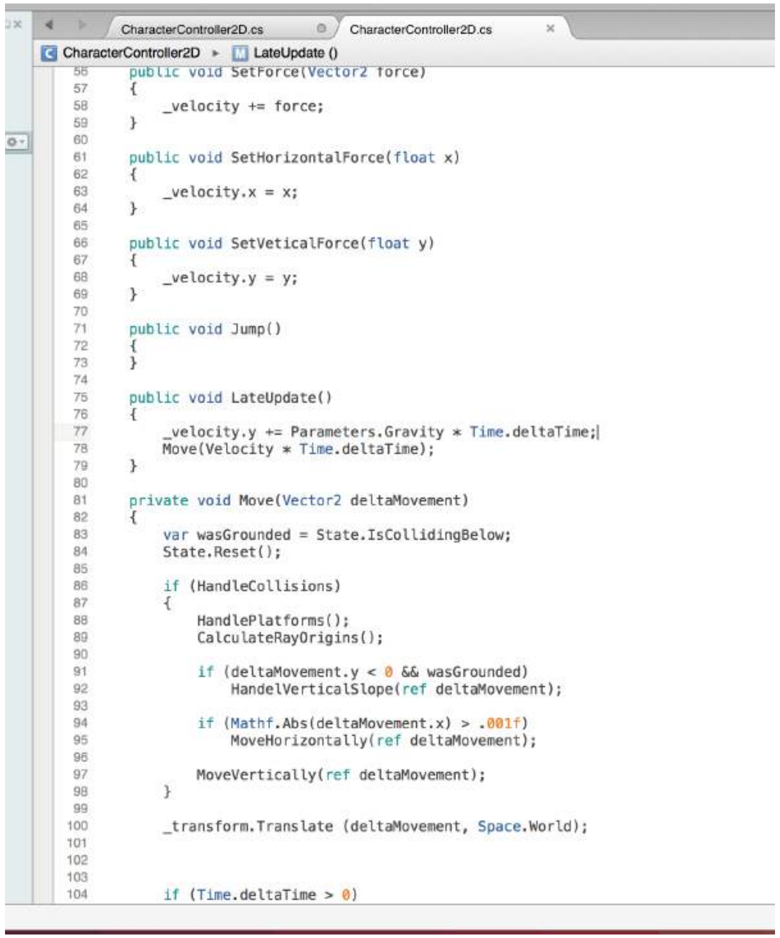


```
private void MoveVertically(ref Vector2 deltaMovement)
{
    var isGoingUp = deltaMovement.y > 0;
    var rayDistance = Mathf.Abs(deltaMovement.y) + SkinWidth;
    var rayDirection = isGoingUp ? Vector2.up : -Vector2.up;
    var rayOrigin = isGoingUp ? _raycastTopLeft : _raycastBottomLeft;

    rayOrigin.x += deltaMovement.x;

    var standingOnDistance = float.MaxValue;
    for (var i = 0; i < TotalVerticalRays; i++)
    {
        var rayVector = new Vector2(rayOrigin.x + (i * _horizontalDistanceBetweenRays), rayOrigin.y);
        Debug.DrawRay(rayVector, rayDirection * rayDistance, Color.red);

        var raycastHit = Physics2D.Raycast(rayVector, rayDirection, rayDistance, PlatformMask);
        if (!raycastHit)
            continue;

        if (!isGoingUp)
        {
            var verticalDistanceToHit = _transform.position.y - raycastHit.point.y;
            if (verticalDistanceToHit < standingOnDistance)
            {
                standingOnDistance = verticalDistanceToHit;
                StandingOn = raycastHit.collider.gameObject;
            }
        }

        deltaMovement.y = raycastHit.point.y - rayVector.y;
        rayDistance = Mathf.Abs(deltaMovement.y);

        if (isGoingUp)
        {
            deltaMovement.y -= SkinWidth;
            State.IsCollidingAbove = true;
        }
        else
        {
            deltaMovement.y += SkinWidth;
            State.IsCollidingBelow = true;
        }

        if (!isGoingUp && deltaMovement.y > .0001f)
            State.IsMovingUpSlope = true;

        if (rayDistance < SkinWidth + .0001f)
            break;
    }
```

GAME DEVELOPMENT

- CHARACTER CONTROLLER

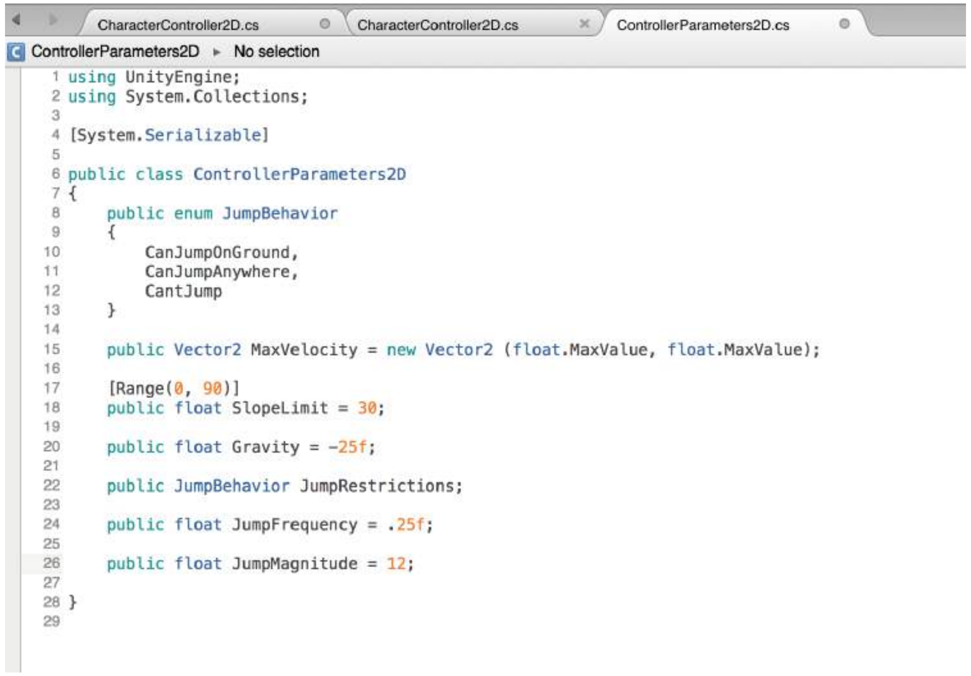
By adding gravity into the game in the late update class the player will fall down or be able to jump (when I implement it soon).



```
    public void SetForce(Vector2 force)
    {
        _velocity += force;
    }

    public void SetHorizontalForce(float x)
    {
        _velocity.x = x;
    }

    public void SetVeticalForce(float y)
    {
        _velocity.y = y;
    }

    public void Jump()
    {
    }

    public void LateUpdate()
    {
        _velocity.y += Parameters.Gravity * Time.deltaTime;
        Move(Velocity * Time.deltaTime);
    }

    private void Move(Vector2 deltaMovement)
    {
        var wasGrounded = State.IsCollidingBelow;
        State.Reset();

        if (HandleCollisions)
        {
            HandlePlatforms();
            CalculateRayOrigins();

            if (deltaMovement.y < 0 && wasGrounded)
                HandelVerticalSlope(ref deltaMovement);

            if (Mathf.Abs(deltaMovement.x) > .001f)
                MoveHorizontally(ref deltaMovement);

            MoveVertically(ref deltaMovement);
        }

        _transform.Translate (deltaMovement, Space.World);


        if (Time.deltaTime > 0)
```

- CHARACTER CONTROLLER

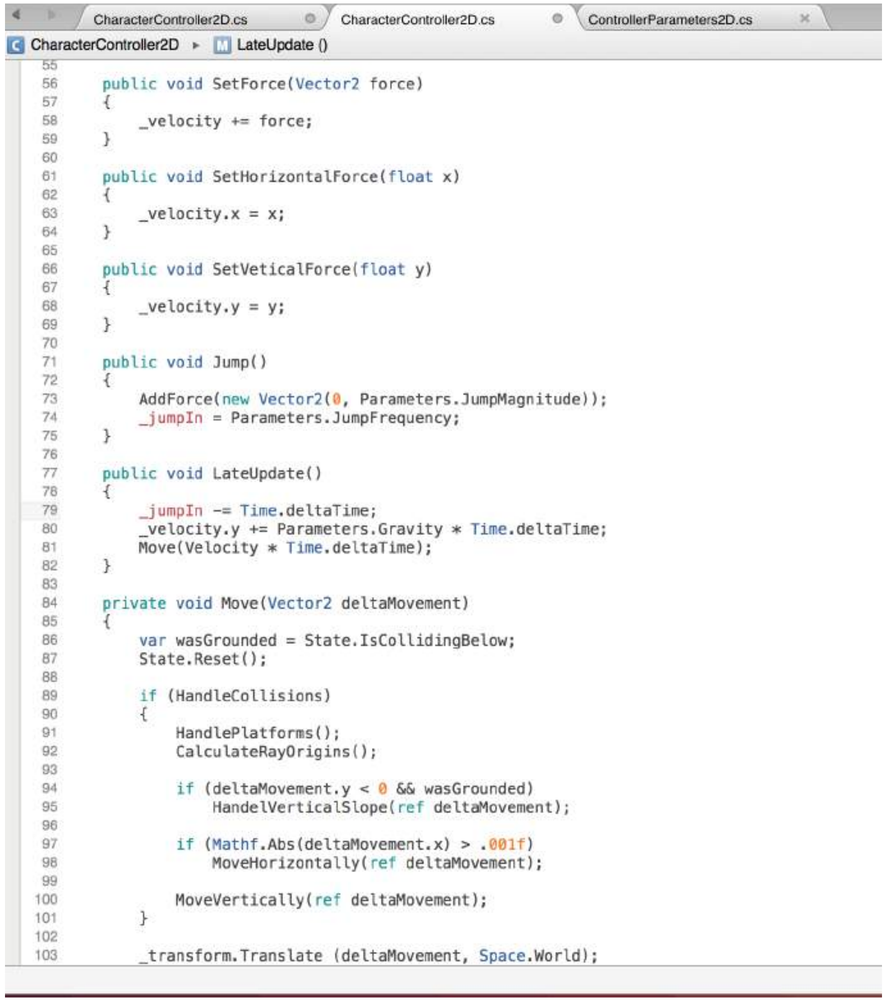
In the 'ControllerParameters2D' class I implement a few classes to show how much velocity will be added to the player when a jump control is performed, so just like in real life, gravity keeps the player grounded but when an input is pressed this will allow the player to jump up to a certain distance then fall back down to the ground.


CharacterController2D.cs
CharacterController2D.cs
ControllerParameters2D.cs
ControllerParameters2D ▸ No selection

```
using UnityEngine;
using System.Collections;

[System.Serializable]

public class ControllerParameters2D
{
    public enum JumpBehavior
    {
        CanJumpOnGround,
        CanJumpAnywhere,
        CantJump
    }

    public Vector2 MaxVelocity = new Vector2 (float.MaxValue, float.MaxValue);

    [Range(0, 90)]
    public float SlopeLimit = 30;

    public float Gravity = -25f;

    public JumpBehavior JumpRestrictions;

    public float JumpFrequency = .25f;

    public float JumpMagnitude = 12;

}
```

- CHARACTER CONTROLLER

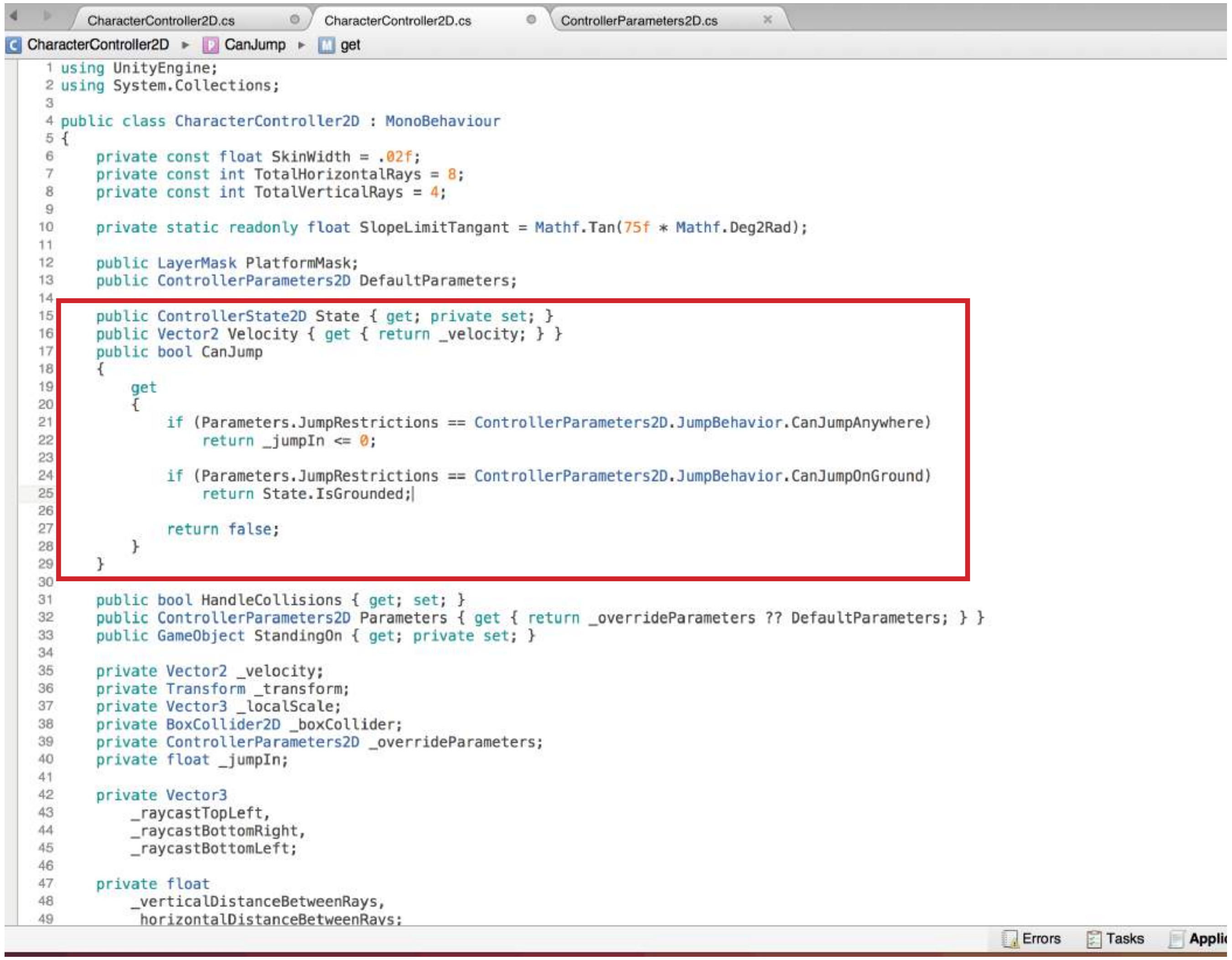
In the jump method in the CharacterController2D script, this code I have added will add force to the player to be able to perform a jump. I just need to add the '_JumpIn' parameter at the stat of the script so it will compile no errors.


CharacterController2D.cs
CharacterController2D.cs
ControllerParameters2D.cs
CharacterController2D ▸ LateUpdate ()

```
    public void SetForce(Vector2 force)
    {
        _velocity += force;
    }

    public void SetHorizontalForce(float x)
    {
        _velocity.x = x;
    }

    public void SetVeticalForce(float y)
    {
        _velocity.y = y;
    }

    public void Jump()
    {
        AddForce(new Vector2(0, Parameters.JumpMagnitude));
        _jumpIn = Parameters.JumpFrequency;
    }

    public void LateUpdate()
    {
        _jumpIn -= Time.deltaTime;
        _velocity.y += Parameters.Gravity * Time.deltaTime;
        Move(Velocity * Time.deltaTime);
    }

    private void Move(Vector2 deltaMovement)
    {
        var wasGrounded = State.IsCollidingBelow;
        State.Reset();

        if (HandleCollisions)
        {
            HandlePlatforms();
            CalculateRayOrigins();

            if (deltaMovement.y < 0 && wasGrounded)
                HandelVerticalSlope(ref deltaMovement);

            if (Mathf.Abs(deltaMovement.x) > .001f)
                MoveHorizontally(ref deltaMovement);

            MoveVertically(ref deltaMovement);
        }

        _transform.Translate (deltaMovement, Space.World);
```

- CHARACTER CONTROLLER

Also in the CharacterController2D script, I have added jump restrictions in so that the character can only jump when on the ground, and not in mid air like a double jump type mechanic because my jump platform I will have in my game will act as the double jump.

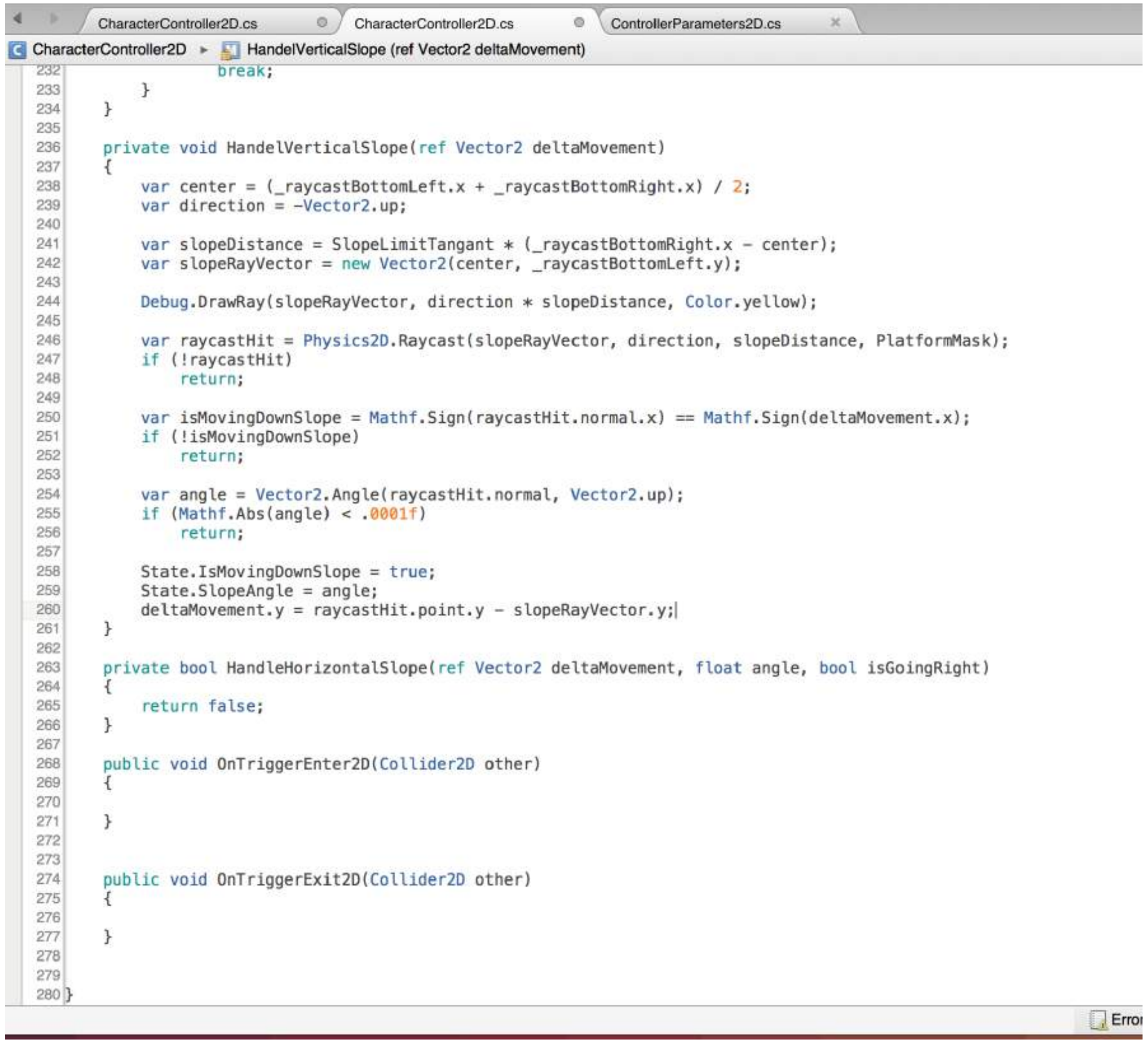


```
using UnityEngine;
using System.Collections;

public class CharacterController2D : MonoBehaviour
{
    private const float SkinWidth = .02f;
    private const int TotalHorizontalRays = 8;
    private const int TotalVerticalRays = 4;

    private static readonly float SlopeLimitTangant = Mathf.Tan(75f * Mathf.Deg2Rad);

    public LayerMask PlatformMask;
    public ControllerParameters2D DefaultParameters;

    public ControllerState2D State { get; private set; }
    public Vector2 Velocity { get { return _velocity; } }
    public bool CanJump
    {
        get
        {
            if (Parameters.JumpRestrictions == ControllerParameters2D.JumpBehavior.CanJumpAnywhere)
                return _jumpIn <= 0;

            if (Parameters.JumpRestrictions == ControllerParameters2D.JumpBehavior.CanJumpOnGround)
                return State.IsGrounded;

            return false;
        }
    }

    public bool HandleCollisions { get; set; }
    public ControllerParameters2D Parameters { get { return _overrideParameters ?? DefaultParameters; } }
    public GameObject StandingOn { get; private set; }

    private Vector2 _velocity;
    private Transform _transform;
    private Vector3 _localScale;
    private BoxCollider2D _boxCollider;
    private ControllerParameters2D _overrideParameters;
    private float _jumpIn;

    private Vector3
        _raycastTopLeft,
        _raycastBottomRight,
        _raycastBottomLeft;

    private float
        _verticalDistanceBetweenRays,
        _horizontalDistanceBetweenRays;
```

- CHARACTER CONTROLLER

In order for my player to not get stuck or stutter on platforms as occasionally while testing it I have noticed slight glitches I have added a public class to handle vertical slopes.

This code calculates the angle of the slope and if the platform is set at a certain angle the gravity will slowly pull the player to the ground smoothly, as if the player was slipping.

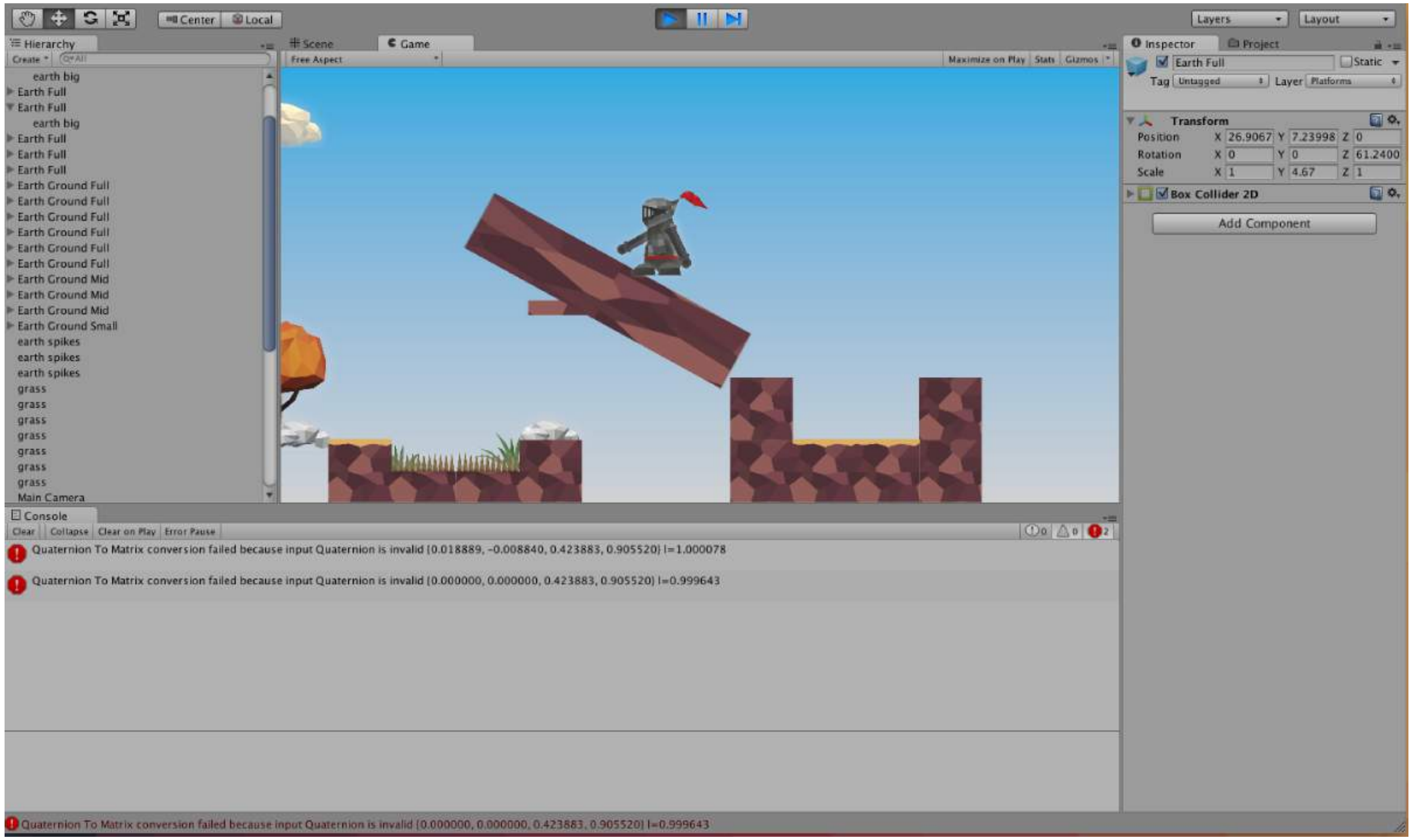

CharacterController2D.cs
CharacterController2D.cs
ControllerParameters2D.cs
CharacterController2D ▸ HandelVerticalSlope (ref Vector2 deltaMovement)
Error

```
                break;
        }
    }

    private void HandelVerticalSlope(ref Vector2 deltaMovement)
    {
        var center = (_raycastBottomLeft.x + _raycastBottomRight.x) / 2;
        var direction = -Vector2.up;

        var slopeDistance = SlopeLimitTangant * (_raycastBottomRight.x - center);
        var slopeRayVector = new Vector2(center, _raycastBottomLeft.y);

        Debug.DrawRay(slopeRayVector, direction * slopeDistance, Color.yellow);

        var raycastHit = Physics2D.Raycast(slopeRayVector, direction, slopeDistance, PlatformMask);
        if (!raycastHit)
            return;

        var isMovingDownSlope = Mathf.Sign(raycastHit.normal.x) == Mathf.Sign(deltaMovement.x);
        if (!isMovingDownSlope)
            return;

        var angle = Vector2.Angle(raycastHit.normal, Vector2.up);
        if (Mathf.Abs(angle) < .0001f)
            return;

        State.IsMovingDownSlope = true;
        State.SlopeAngle = angle;
        deltaMovement.y = raycastHit.point.y - slopeRayVector.y;
    }

    private bool HandleHorizontalSlope(ref Vector2 deltaMovement, float angle, bool isGoingRight)
    {
        return false;
    }

    public void OnTriggerEnter2D(Collider2D other)
    {

    }


    public void OnTriggerExit2D(Collider2D other)
    {

    }


}
```

- CHARACTER CONTROLLER

In Unity, I place a platform and stretch it slightly (note I wont make it look like this is in the final version, this is for test purposes only).

When the player is idle the player will slide down the slope really smoothly and getting back up is a little bit more difficult which is the effect that I am after.



Earth Full — Static
Tag: Untagged — Layer: Platforms

Transform
Position X 26.9067 Y 7.23998 Z 0
Rotation X 0 Y 0 Z 61.2400
Scale X 1 Y 4.67 Z 1

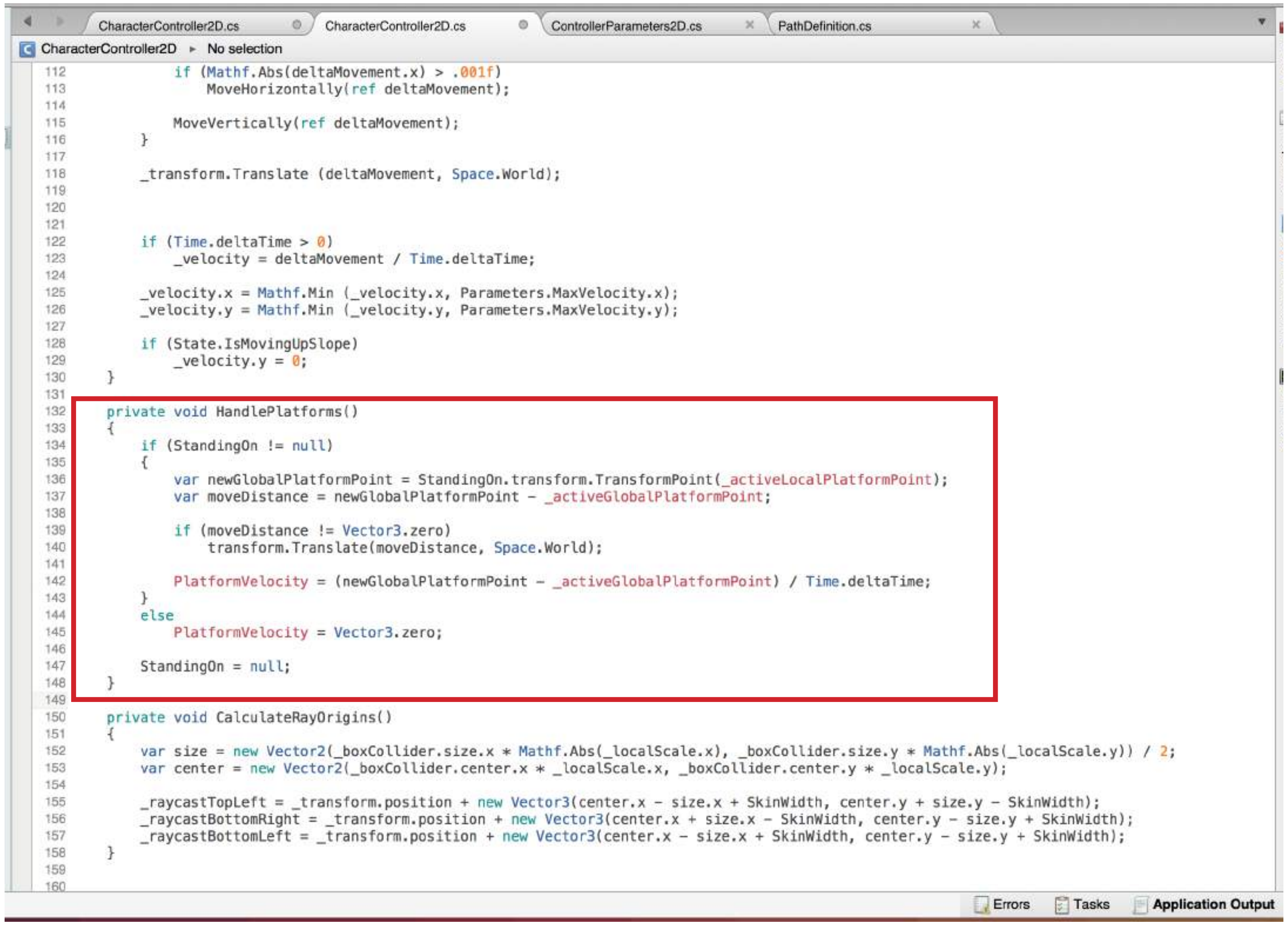
Box Collider 2D

```
Quaternion To Matrix conversion failed because input Quaternion is invalid {0.018889, -0.008840, 0.423883, 0.905520} l=1.000078
Quaternion To Matrix conversion failed because input Quaternion is invalid {0.000000, 0.000000, 0.423883, 0.905520} l=0.999643
Quaternion To Matrix conversion failed because input Quaternion is invalid {0.000000, 0.000000, 0.423883, 0.905520} l=0.999643
```

- CHARACTER CONTROLLER

As I have said I want to include jump platforms that will act as a kind of double jump mechanic in my game.

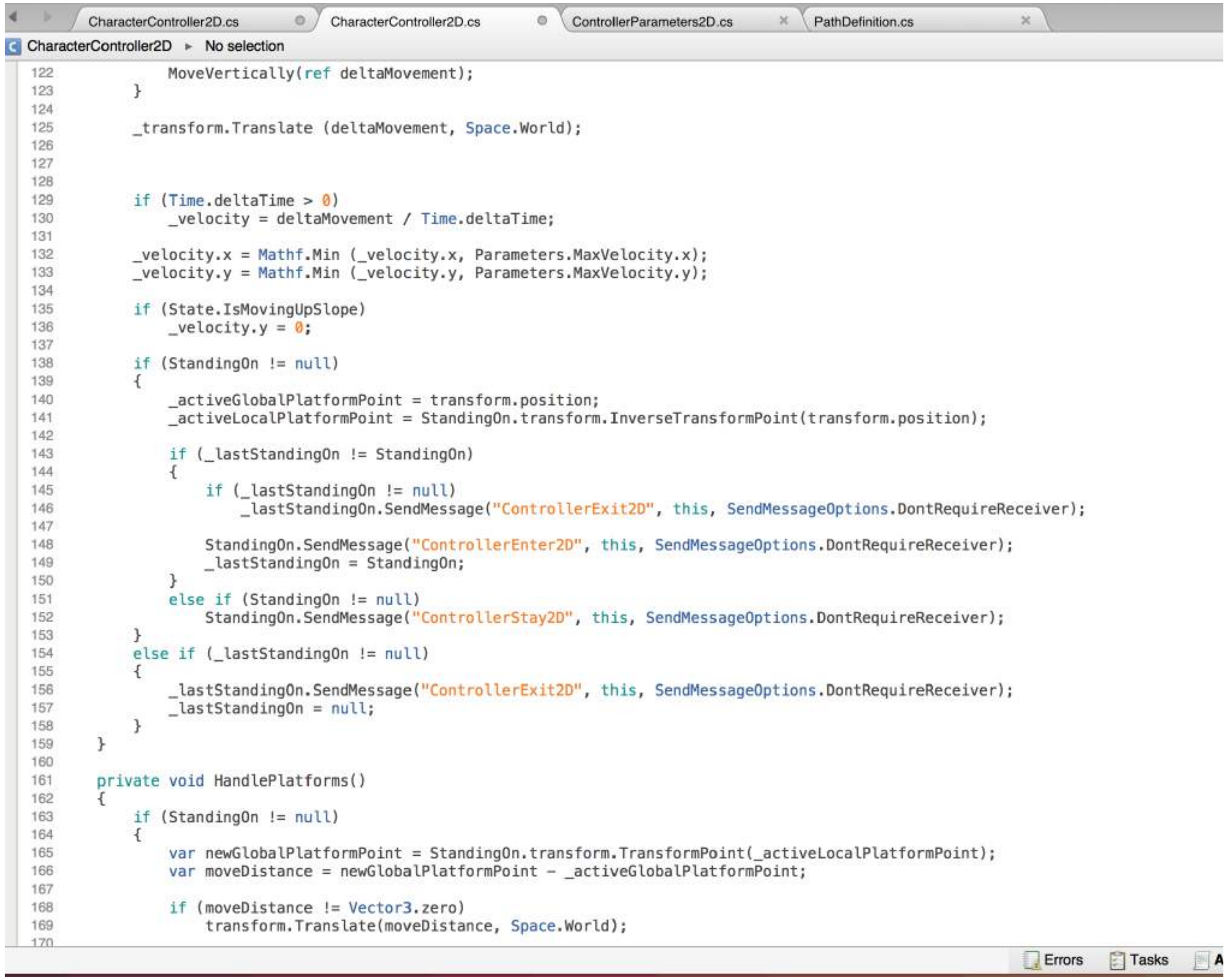
In the CharacterController2D script I fill out the 'HandlePlatforms' class. This will be active when a player stands on to a special platform and there is velocity added to it and will eject the player into the air.



```
                if (Mathf.Abs(deltaMovement.x) > .001f)
                    MoveHorizontally(ref deltaMovement);

                MoveVertically(ref deltaMovement);
            }

            _transform.Translate (deltaMovement, Space.World);



            if (Time.deltaTime > 0)
                _velocity = deltaMovement / Time.deltaTime;

            _velocity.x = Mathf.Min (_velocity.x, Parameters.MaxVelocity.x);
            _velocity.y = Mathf.Min (_velocity.y, Parameters.MaxVelocity.y);

            if (State.IsMovingUpSlope)
                _velocity.y = 0;
        }

        private void HandlePlatforms()
        {
            if (StandingOn != null)
            {
                var newGlobalPlatformPoint = StandingOn.transform.TransformPoint(_activeLocalPlatformPoint);
                var moveDistance = newGlobalPlatformPoint - _activeGlobalPlatformPoint;

                if (moveDistance != Vector3.zero)
                    transform.Translate(moveDistance, Space.World);

                PlatformVelocity = (newGlobalPlatformPoint - _activeGlobalPlatformPoint) / Time.deltaTime;
            }
            else
                PlatformVelocity = Vector3.zero;

            StandingOn = null;
        }

        private void CalculateRayOrigins()
        {
            var size = new Vector2(_boxCollider.size.x * Mathf.Abs(_localScale.x), _boxCollider.size.y * Mathf.Abs(_localScale.y)) / 2;
            var center = new Vector2(_boxCollider.center.x * _localScale.x, _boxCollider.center.y * _localScale.y);

            _raycastTopLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y + size.y - SkinWidth);
            _raycastBottomRight = _transform.position + new Vector3(center.x + size.x - SkinWidth, center.y - size.y + SkinWidth);
            _raycastBottomLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y - size.y + SkinWidth);
        }
```

- CHARACTER CONTROLLER

While standing on the platform this code I have implemented will send messages within Unity to make the player jump, speed up or do whatever I want it to.

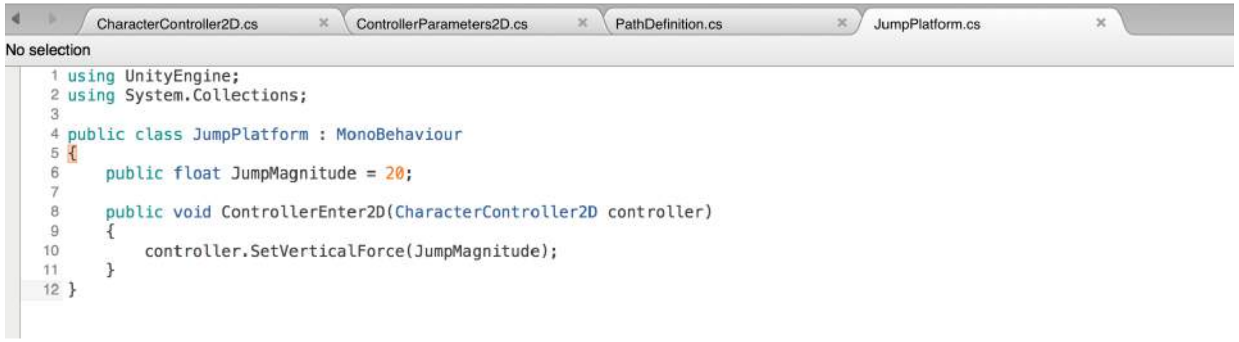


```
        MoveVertically(ref deltaMovement);
    }

    _transform.Translate (deltaMovement, Space.World);


    if (Time.deltaTime > 0)
        _velocity = deltaMovement / Time.deltaTime;

    _velocity.x = Mathf.Min (_velocity.x, Parameters.MaxVelocity.x);
    _velocity.y = Mathf.Min (_velocity.y, Parameters.MaxVelocity.y);

    if (State.IsMovingUpSlope)
        _velocity.y = 0;

    if (StandingOn != null)
    {
        _activeGlobalPlatformPoint = transform.position;
        _activeLocalPlatformPoint = StandingOn.transform.InverseTransformPoint(transform.position);

        if (_lastStandingOn != StandingOn)
        {
            if (_lastStandingOn != null)
                _lastStandingOn.SendMessage("ControllerExit2D", this, SendMessageOptions.DontRequireReceiver);

            StandingOn.SendMessage("ControllerEnter2D", this, SendMessageOptions.DontRequireReceiver);
            _lastStandingOn = StandingOn;
        }
        else if (StandingOn != null)
            StandingOn.SendMessage("ControllerStay2D", this, SendMessageOptions.DontRequireReceiver);
    }
    else if (_lastStandingOn != null)
    {
        _lastStandingOn.SendMessage("ControllerExit2D", this, SendMessageOptions.DontRequireReceiver);
        _lastStandingOn = null;
    }
}

private void HandlePlatforms()
{
    if (StandingOn != null)
    {
        var newGlobalPlatformPoint = StandingOn.transform.TransformPoint(_activeLocalPlatformPoint);
        var moveDistance = newGlobalPlatformPoint - _activeGlobalPlatformPoint;

        if (moveDistance != Vector3.zero)
            transform.Translate(moveDistance, Space.World);
```

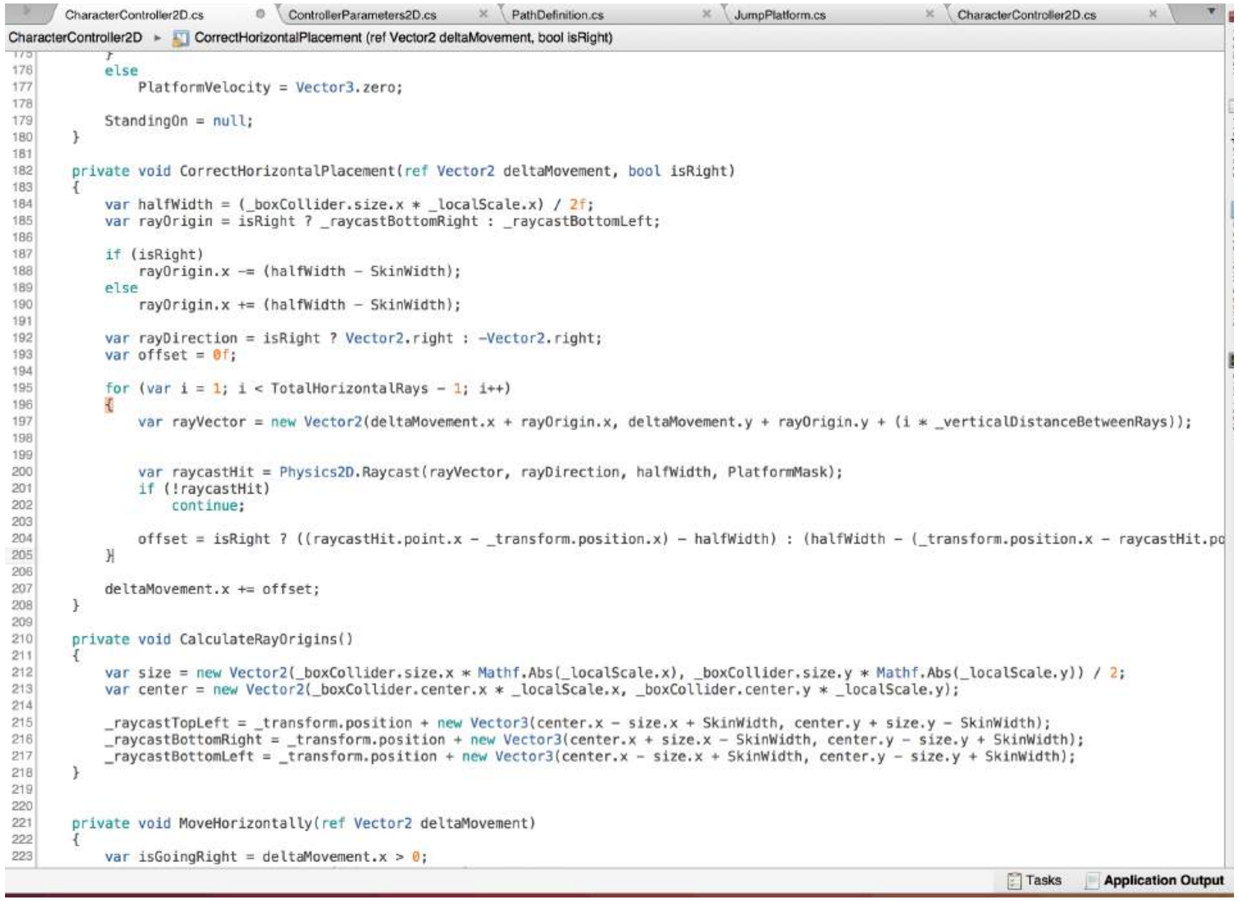
- CHARACTER CONTROLLER

I created a new c# script for jumping platforms that give the player more magnitude (This is the script that will be assigned to the special platform).

Back in the CharacterController2D script I added new code to push player to the left or right. If they hit a moving platform while being ejected by the jump platform they no longer float in a platform and get stuck. Now, the user gets repelled away from the platform.



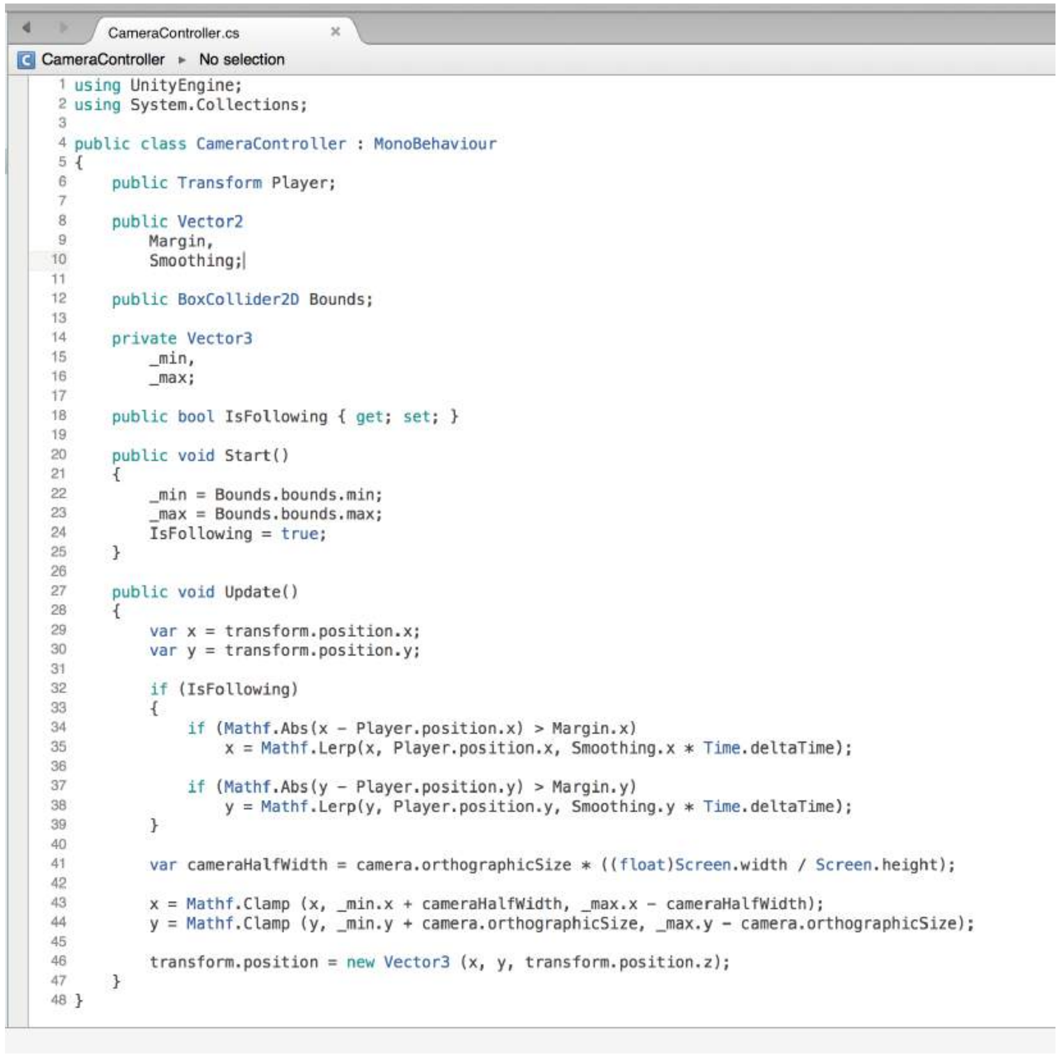
```csharp
using UnityEngine;
using System.Collections;

public class JumpPlatform : MonoBehaviour
{
    public float JumpMagnitude = 20;

    public void ControllerEnter2D(CharacterController2D controller)
    {
        controller.SetVerticalForce(JumpMagnitude);
    }
}
```


CharacterController2D.cs
ControllerParameters2D.cs
PathDefinition.cs
JumpPlatform.cs
CharacterController2D.cs
CharacterController2D ▸ CorrectHorizontalPlacement (ref Vector2 deltaMovement, bool isRight)
Tasks
Application Output

```csharp
        else
            PlatformVelocity = Vector3.zero;

        StandingOn = null;
    }

    private void CorrectHorizontalPlacement(ref Vector2 deltaMovement, bool isRight)
    {
        var halfWidth = (_boxCollider.size.x * _localScale.x) / 2f;
        var rayOrigin = isRight ? _raycastBottomRight : _raycastBottomLeft;

        if (isRight)
            rayOrigin.x -= (halfWidth - SkinWidth);
        else
            rayOrigin.x += (halfWidth - SkinWidth);

        var rayDirection = isRight ? Vector2.right : -Vector2.right;
        var offset = 0f;

        for (var i = 1; i < TotalHorizontalRays - 1; i++)
        {
            var rayVector = new Vector2(deltaMovement.x + rayOrigin.x, deltaMovement.y + rayOrigin.y + (i * _verticalDistanceBetweenRays));


            var raycastHit = Physics2D.Raycast(rayVector, rayDirection, halfWidth, PlatformMask);
            if (!raycastHit)
                continue;

            offset = isRight ? ((raycastHit.point.x - _transform.position.x) - halfWidth) : (halfWidth - (_transform.position.x - raycastHit.po
        }

        deltaMovement.x += offset;
    }

    private void CalculateRayOrigins()
    {
        var size = new Vector2(_boxCollider.size.x * Mathf.Abs(_localScale.x), _boxCollider.size.y * Mathf.Abs(_localScale.y)) / 2;
        var center = new Vector2(_boxCollider.center.x * _localScale.x, _boxCollider.center.y * _localScale.y);

        _raycastTopLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y + size.y - SkinWidth);
        _raycastBottomRight = _transform.position + new Vector3(center.x + size.x - SkinWidth, center.y - size.y + SkinWidth);
        _raycastBottomLeft = _transform.position + new Vector3(center.x - size.x + SkinWidth, center.y - size.y + SkinWidth);
    }


    private void MoveHorizontally(ref Vector2 deltaMovement)
    {
        var isGoingRight = deltaMovement.x > 0;
```

- CAMERA CONTROLLER

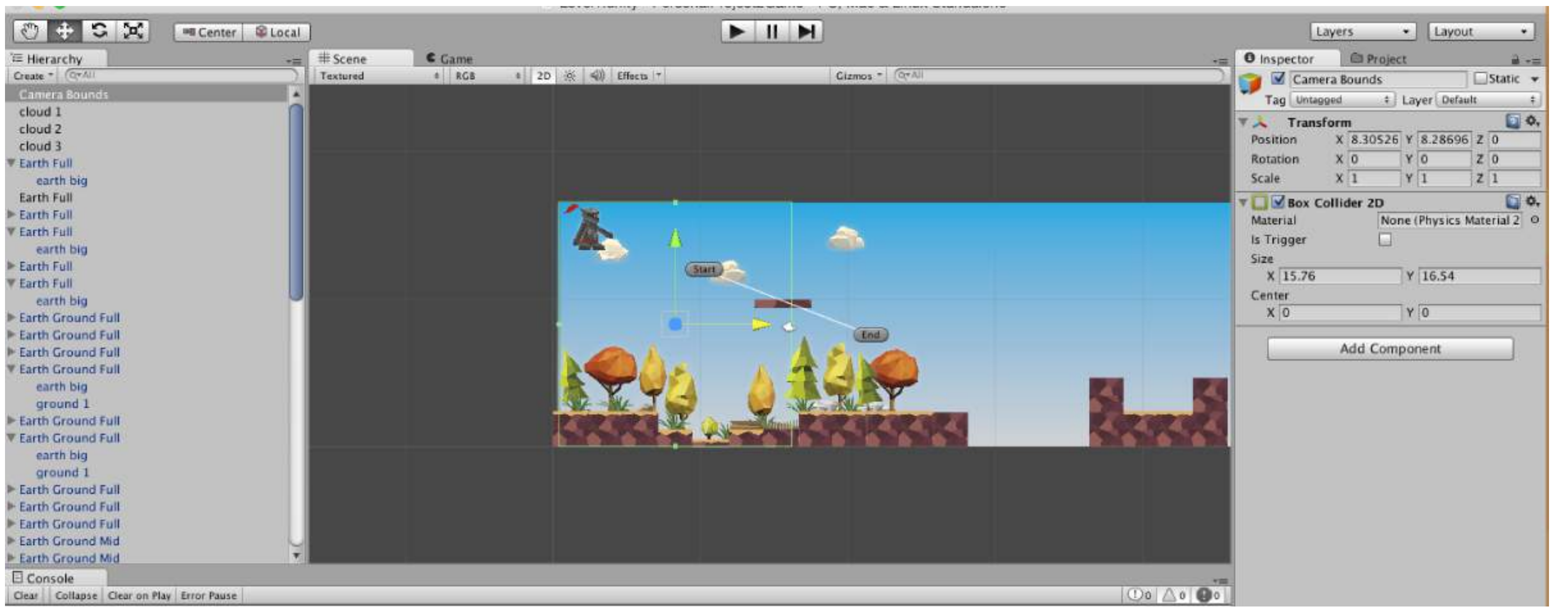
I want the camera to follow the player along the level because if it were a static camera, in that window, the level would be tiny. The code here tells Unity basically, that wherever the player is on screen, the camera will follow from the center of its bounding box (which I create on the next page).



```
using UnityEngine;
using System.Collections;

public class CameraController : MonoBehaviour
{
    public Transform Player;

    public Vector2
        Margin,
        Smoothing;

    public BoxCollider2D Bounds;

    private Vector3
        _min,
        _max;

    public bool IsFollowing { get; set; }

    public void Start()
    {
        _min = Bounds.bounds.min;
        _max = Bounds.bounds.max;
        IsFollowing = true;
    }

    public void Update()
    {
        var x = transform.position.x;
        var y = transform.position.y;

        if (IsFollowing)
        {
            if (Mathf.Abs(x - Player.position.x) > Margin.x)
                x = Mathf.Lerp(x, Player.position.x, Smoothing.x * Time.deltaTime);

            if (Mathf.Abs(y - Player.position.y) > Margin.y)
                y = Mathf.Lerp(y, Player.position.y, Smoothing.y * Time.deltaTime);
        }

        var cameraHalfWidth = camera.orthographicSize * ((float)Screen.width / Screen.height);

        x = Mathf.Clamp (x, _min.x + cameraHalfWidth, _max.x - cameraHalfWidth);
        y = Mathf.Clamp (y, _min.y + camera.orthographicSize, _max.y - camera.orthographicSize);

        transform.position = new Vector3 (x, y, transform.position.z);
    }
}
```

- CAMERA CONTROLLER

Creating a bounding box for the camera is pretty straight forward, I create a new game object and attach a Box Collider 2D to it. This new object has to fill the level at a reasonable size, in the image to the right the box with the green stroke represents how big my camera bounds is for the level, so the camera will be focusing on what is in the box at all times, this will be the player, running through the level.



Camera Bounds
cloud 1
cloud 2
cloud 3
Earth Full
earth big
Earth Full
Earth Full
Earth Full
earth big
Earth Full
Earth Full
earth big
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Full
earth big
ground 1
Earth Ground Full
Earth Ground Full
earth big
ground 1
Earth Ground Full
Earth Ground Full
Earth Ground Full
Earth Ground Mid
Earth Ground Mid

Camera Bounds [✓] Static [ ]
Tag Untagged Layer Default

Transform
Position X 8.30526 Y 8.28696 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Box Collider 2D [✓]
Material None (Physics Material 2
Is Trigger [ ]
Size
X 15.76 Y 16.54
Center
X 0 Y 0

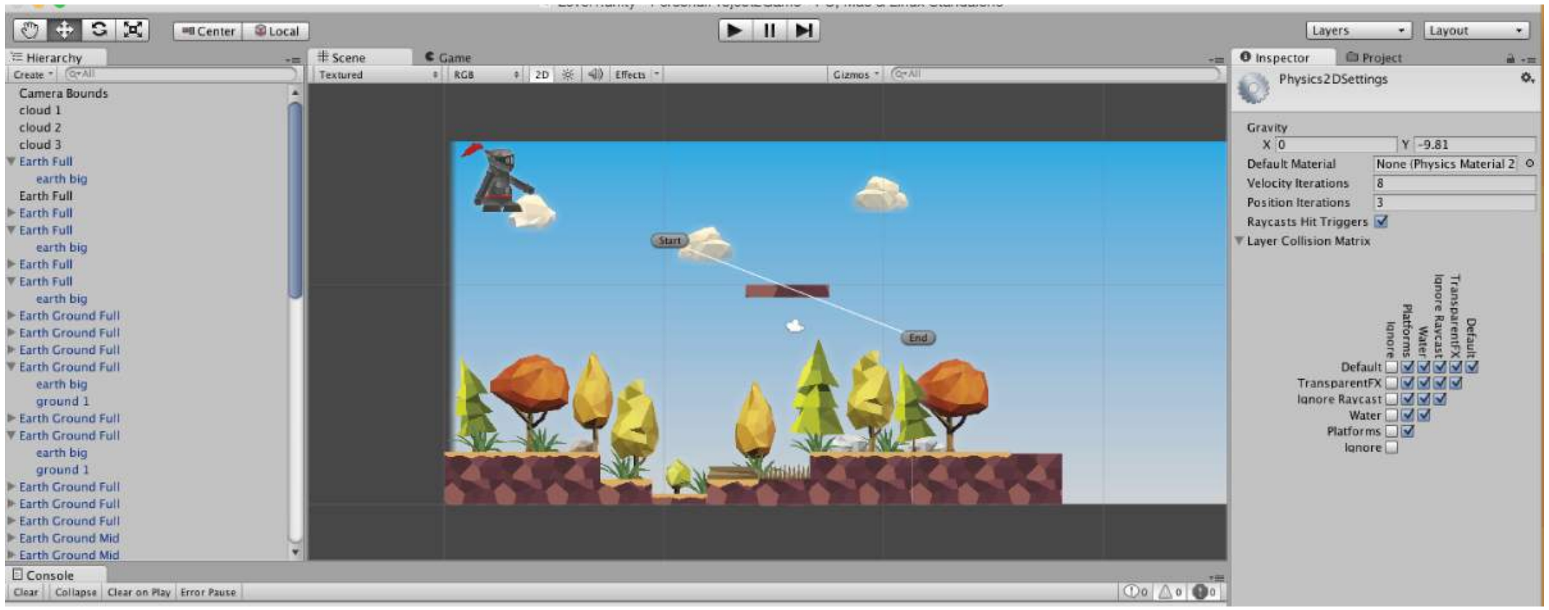
GAME DEVELOPMENT

- CAMERA CONTROLLER

I need to make sure my player does not collide with this box collider, I need to put it on a new layer and make it almost invisible so that it does not interfere with and of the game objects.

I make a new layer called "Ignore" and unchecked everything for that layer so nothing will collide with the camera bounds game object in the Physics 2D settings menu.

Once this has been implemented, the camera follows the player and I also adjusted camera to zoom into scene of game. Because I have done this the overall game experience is coming together nicely, it feels responsive.



```
Camera Bounds
cloud 1
cloud 2
cloud 3
▼ Earth Full
    earth big
Earth Full
▶ Earth Full
▼ Earth Full
    earth big
▶ Earth Full
▼ Earth Full
    earth big
▶ Earth Ground Full
▶ Earth Ground Full
▶ Earth Ground Full
▼ Earth Ground Full
    earth big
    ground 1
▶ Earth Ground Full
▼ Earth Ground Full
    earth big
    ground 1
▶ Earth Ground Full
▶ Earth Ground Full
▶ Earth Ground Full
▶ Earth Ground Mid
▶ Earth Ground Mid
```

Physics2DSettings

Gravity	X 0 Y -9.81
Default Material	None (Physics Material 2
Velocity Iterations	8
Position Iterations	3
Raycasts Hit Triggers	✓

Layer Collision Matrix

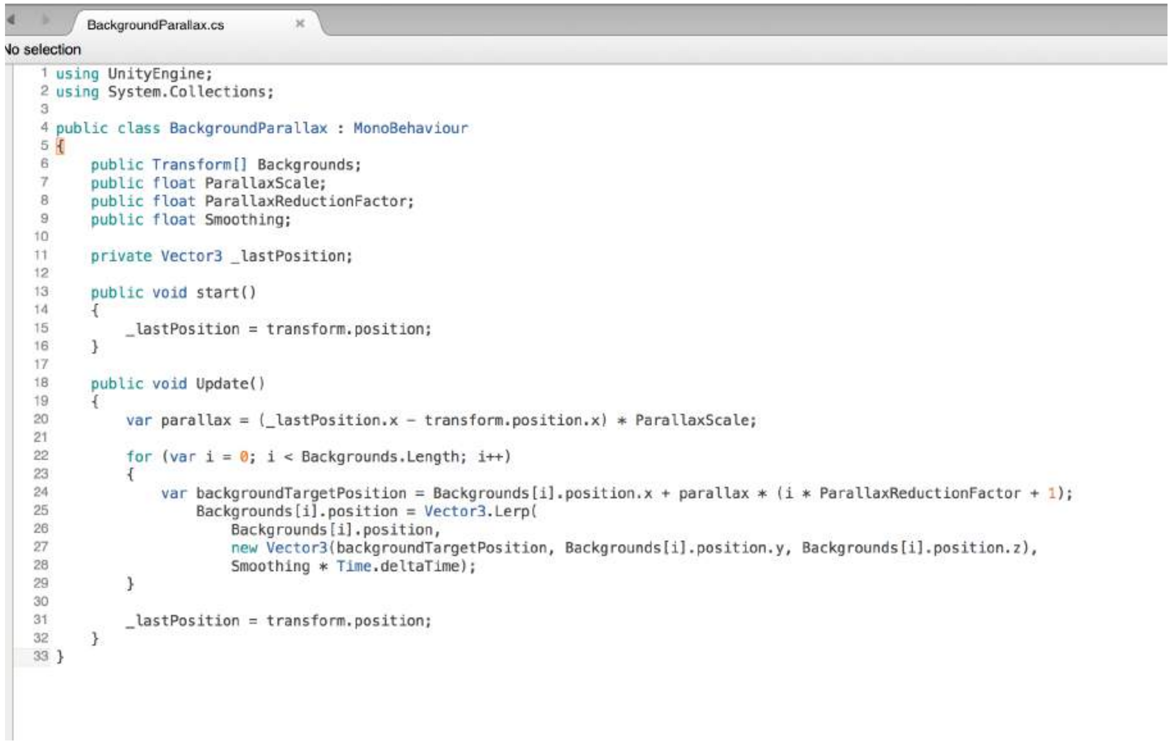
	Ignore	Platforms	Water	Ignore Raycast	TransparentFX	Default
Default	☐	✓	✓	✓	✓	✓
TransparentFX	☐	✓	✓	✓	✓	
Ignore Raycast	☐	✓	✓	✓		
Water	☐	✓	✓			
Platforms	☐	✓				
Ignore	☐					

GAME DEVELOPMENT

- BACKGROUND PARALLAX

For my game to have a little more polish, I want to add a background parallax to its, this will make objects in the background move slower or faster depending on the camera movement.

The code I have written basically does just that on 3 layers (The mountains, trees and clouds game objects). When the player moves, the camera moves, so when the camera moves these selected objects will move at different speeds through the level.

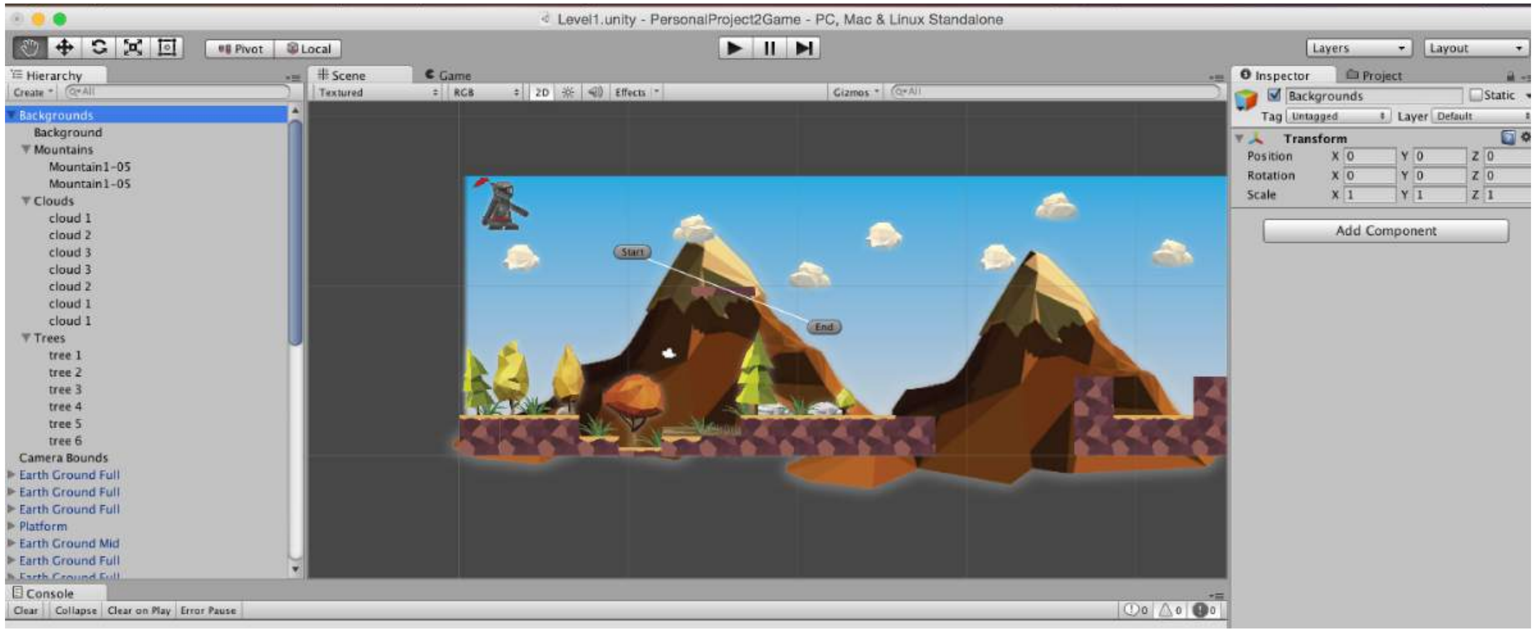


```
using UnityEngine;
using System.Collections;

public class BackgroundParallax : MonoBehaviour
{
    public Transform[] Backgrounds;
    public float ParallaxScale;
    public float ParallaxReductionFactor;
    public float Smoothing;

    private Vector3 _lastPosition;

    public void start()
    {
        _lastPosition = transform.position;
    }

    public void Update()
    {
        var parallax = (_lastPosition.x - transform.position.x) * ParallaxScale;

        for (var i = 0; i < Backgrounds.Length; i++)
        {
            var backgroundTargetPosition = Backgrounds[i].position.x + parallax * (i * ParallaxReductionFactor + 1);
            Backgrounds[i].position = Vector3.Lerp(
                Backgrounds[i].position,
                new Vector3(backgroundTargetPosition, Backgrounds[i].position.y, Backgrounds[i].position.z),
                Smoothing * Time.deltaTime);
        }

        _lastPosition = transform.position;
    }
}
```

- BACKGROUND PARALLAX

Back in Unity I created new game object called Backgrounds, this will be the parent object of the parallax effect. The trees, mountains and clouds are placed in the child objects that are going to be parallax. So as you see the correct objects are attached in the Background slayer.

After this has been done, I press play and the parallax effect is working perfectly, the mountains move the slowest, where the trees move a lot faster when the player moves.



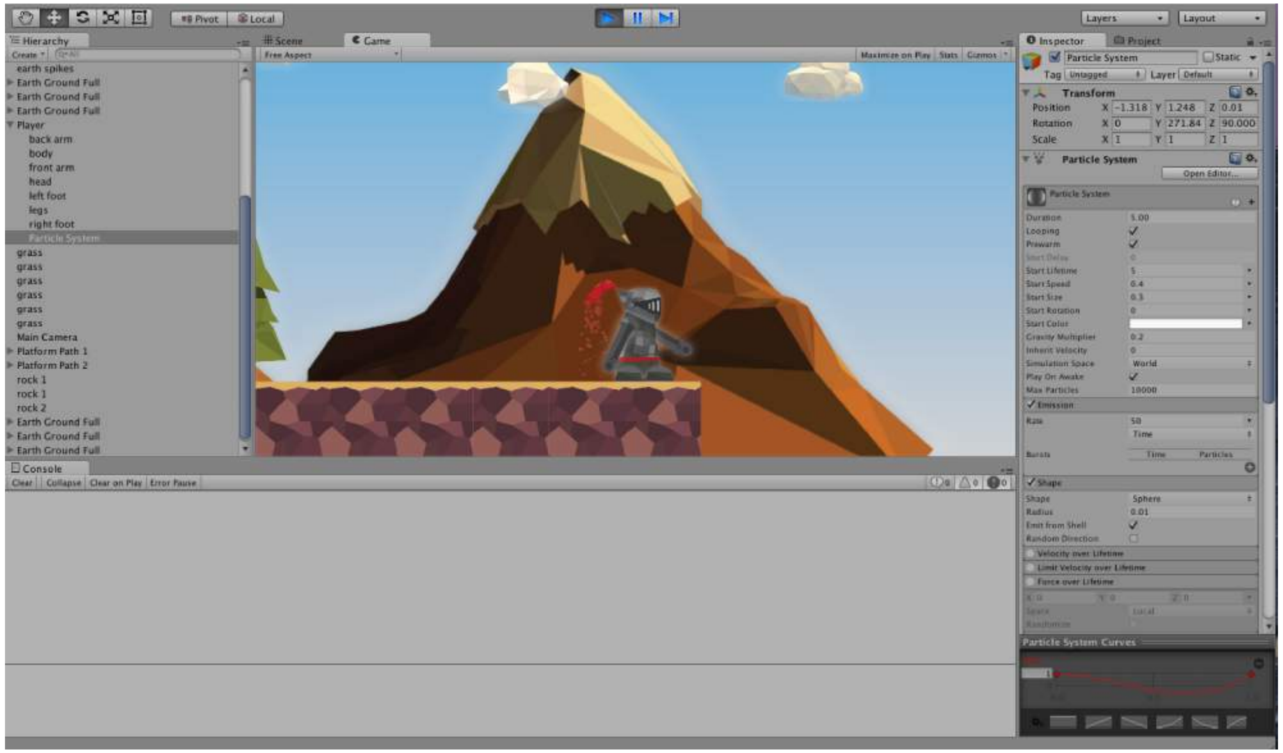
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

GAME DEVELOPMENT

- PARTICLE EFFECTS

For another bit of polish to my game, I am adding a particle system for the player movement, so when the player moves it leaves a trail of particles for visual effect.

I created a new particle system and once I was happy with the colours and the way the gravity dragged the particles to the ground, I added it as a child to the Player layer so it would stay at that position and follow the player.

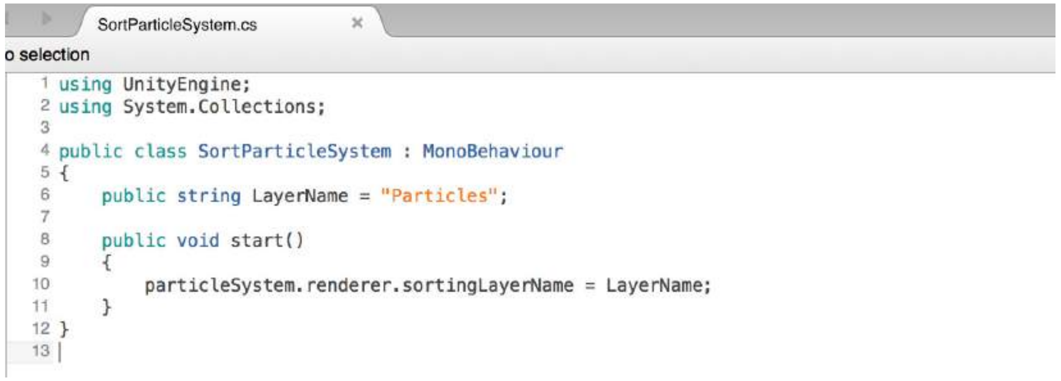


```
earth spikes
▶ Earth Ground Full
▶ Earth Ground Full
▶ Earth Ground Full
▼ Player
    back arm
    body
    front arm
    head
    left foot
    legs
    right foot
    Particle System
grass
grass
grass
grass
grass
grass
Main Camera
▶ Platform Path 1
▶ Platform Path 2
rock 1
rock 1
rock 2
▶ Earth Ground Full
▶ Earth Ground Full
▶ Earth Ground Full
```

```
Particle System          Static
Tag Untagged   Layer Default
Transform
Position   X -1.318  Y 1.248   Z 0.01
Rotation   X 0       Y 271.84  Z 90.000
Scale      X 1       Y 1       Z 1
Particle System
Duration             5.00
Looping              ✓
Prewarm              ✓
Start Delay          0
Start Lifetime       5
Start Speed          0.4
Start Size           0.3
Start Rotation       0
Start Color
Gravity Multiplier   0.2
Inherit Velocity     0
Simulation Space     World
Play On Awake        ✓
Max Particles        10000
✓ Emission
Rate                 50
                     Time
Bursts               Time    Particles
✓ Shape
Shape                Sphere
Radius               0.01
Emit from Shell      ✓
Random Direction     ☐
Velocity over Lifetime
Limit Velocity over Lifetime
Force over Lifetime
```

- PARTICLE EFFECTS

Now that the particle effect is attached to the player, it appears in front of it, which is not really what I want. I need to sort the particles behind player but in front of backgrounds layer. This script does just that.



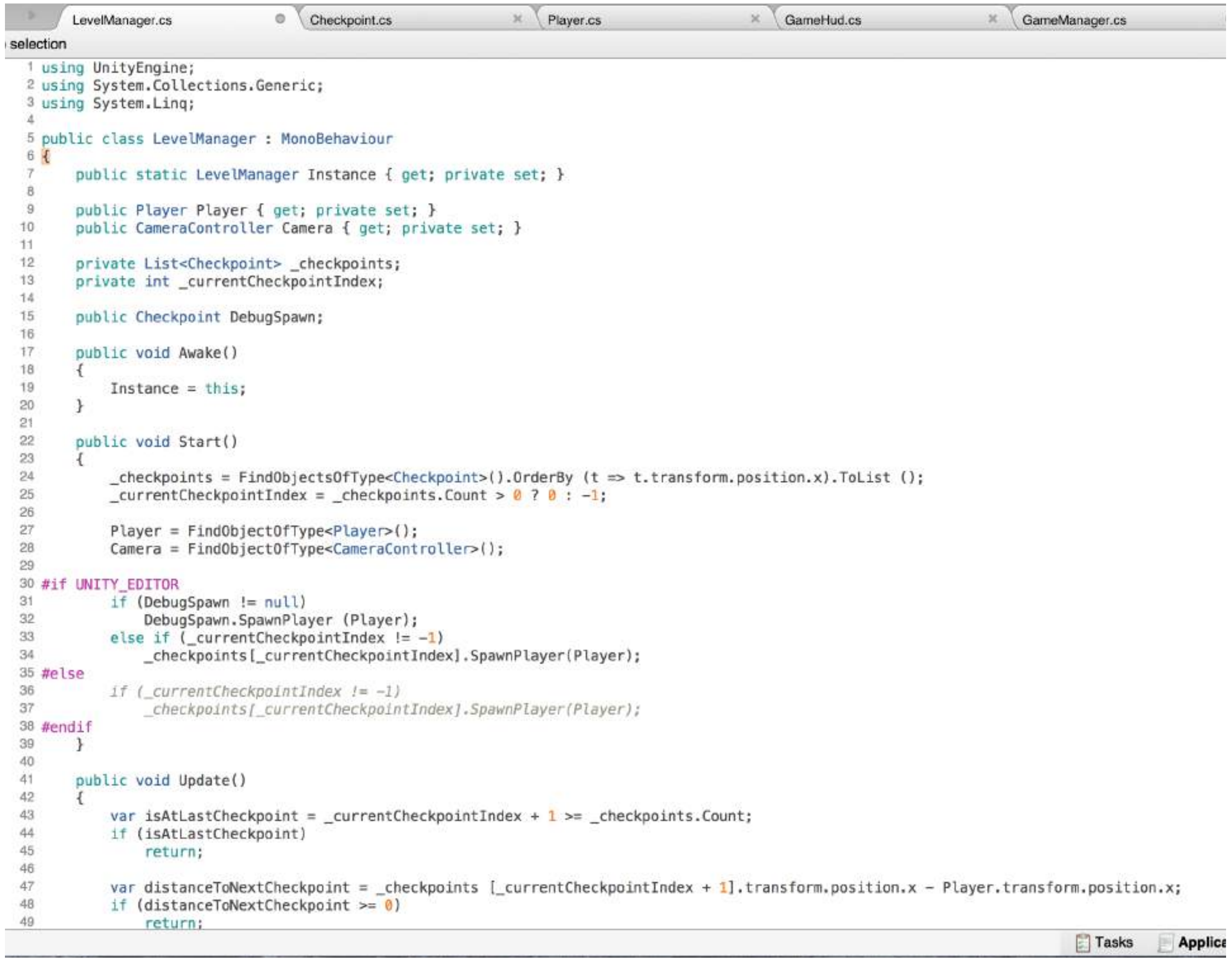
```
using UnityEngine;
using System.Collections;

public class SortParticleSystem : MonoBehaviour
{
    public string LayerName = "Particles";

    public void start()
    {
        particleSystem.renderer.sortingLayerName = LayerName;
    }
}
```

- LEVEL MANAGER

So far we have elements of a game like a character controller but what I have created does not really have any objectives like checkpoints etc.

The level manager controls points and the death of the player. The image here is part 1 of the code (part 2 overleaf).

This basically says that once the game has started, you have 0 points and when you die you respawn at the last known checkpoint with those points will be deducted.

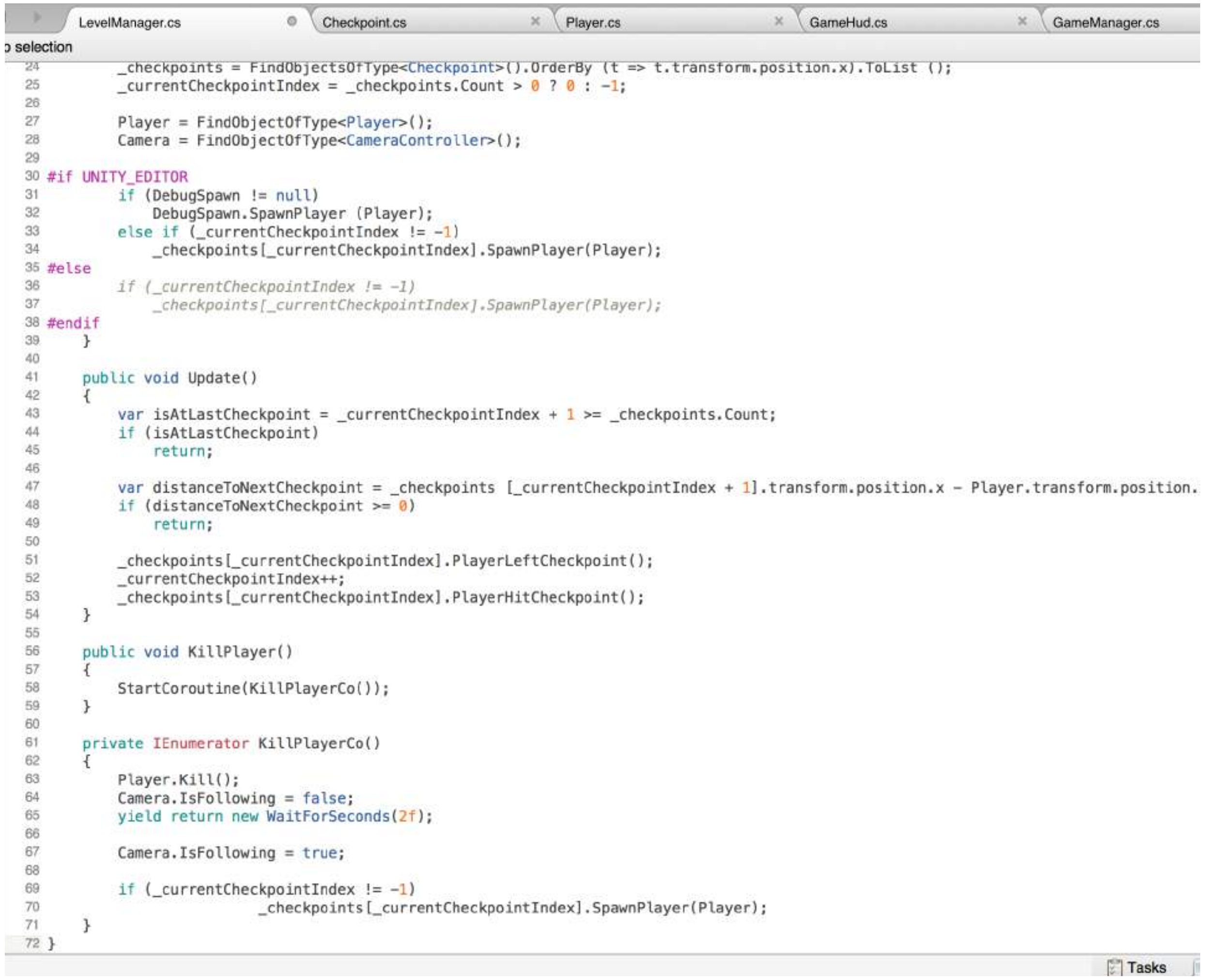


```
using UnityEngine;
using System.Collections.Generic;
using System.Linq;

public class LevelManager : MonoBehaviour
{
    public static LevelManager Instance { get; private set; }

    public Player Player { get; private set; }
    public CameraController Camera { get; private set; }

    private List<Checkpoint> _checkpoints;
    private int _currentCheckpointIndex;

    public Checkpoint DebugSpawn;

    public void Awake()
    {
        Instance = this;
    }

    public void Start()
    {
        _checkpoints = FindObjectsOfType<Checkpoint>().OrderBy (t => t.transform.position.x).ToList ();
        _currentCheckpointIndex = _checkpoints.Count > 0 ? 0 : -1;

        Player = FindObjectOfType<Player>();
        Camera = FindObjectOfType<CameraController>();

#if UNITY_EDITOR
        if (DebugSpawn != null)
            DebugSpawn.SpawnPlayer (Player);
        else if (_currentCheckpointIndex != -1)
            _checkpoints[_currentCheckpointIndex].SpawnPlayer(Player);
#else
        if (_currentCheckpointIndex != -1)
            _checkpoints[_currentCheckpointIndex].SpawnPlayer(Player);
#endif
    }

    public void Update()
    {
        var isAtLastCheckpoint = _currentCheckpointIndex + 1 >= _checkpoints.Count;
        if (isAtLastCheckpoint)
            return;

        var distanceToNextCheckpoint = _checkpoints [_currentCheckpointIndex + 1].transform.position.x - Player.transform.position.x;
        if (distanceToNextCheckpoint >= 0)
            return;
```

GAME DEVELOPMENT

- LEVEL MANAGER

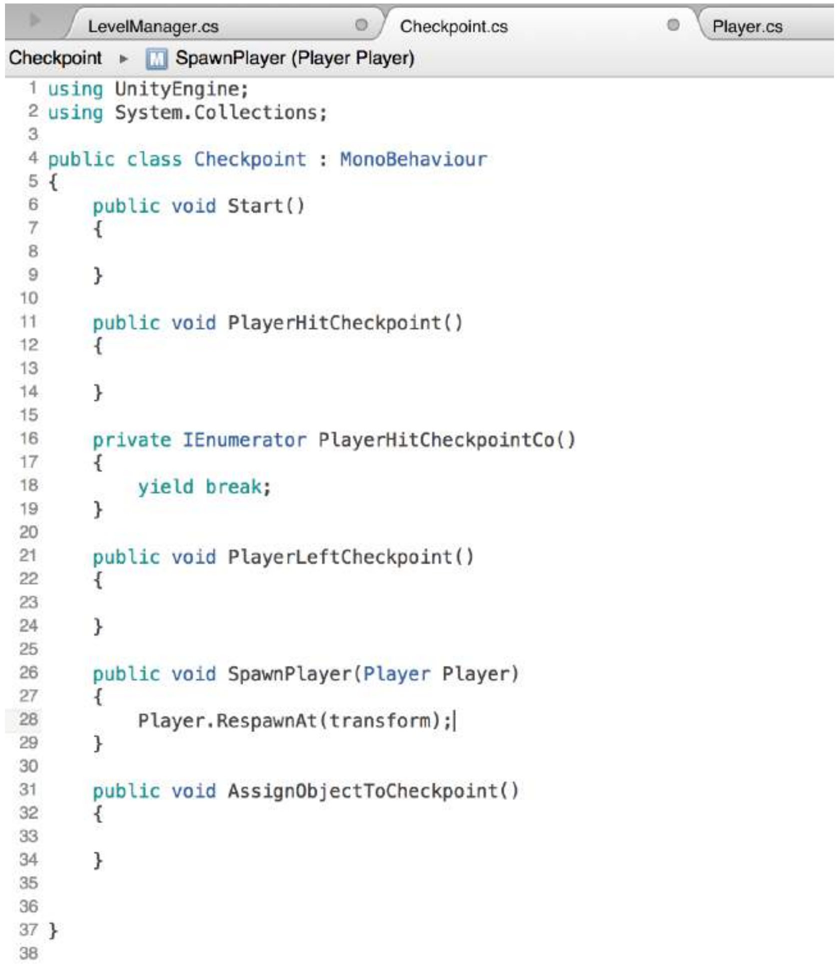
This is part 2 of the level manager code, when the player dies, there's is a 2 second respawn rate and the player will respawn back to normal, with the camera still following the players current position.



```
        _checkpoints = FindObjectsOfType<Checkpoint>().OrderBy (t => t.transform.position.x).ToList ();
        _currentCheckpointIndex = _checkpoints.Count > 0 ? 0 : -1;

        Player = FindObjectOfType<Player>();
        Camera = FindObjectOfType<CameraController>();

#if UNITY_EDITOR
        if (DebugSpawn != null)
            DebugSpawn.SpawnPlayer (Player);
        else if (_currentCheckpointIndex != -1)
            _checkpoints[_currentCheckpointIndex].SpawnPlayer(Player);
#else
        if (_currentCheckpointIndex != -1)
            _checkpoints[_currentCheckpointIndex].SpawnPlayer(Player);
#endif
    }

    public void Update()
    {
        var isAtLastCheckpoint = _currentCheckpointIndex + 1 >= _checkpoints.Count;
        if (isAtLastCheckpoint)
            return;

        var distanceToNextCheckpoint = _checkpoints [_currentCheckpointIndex + 1].transform.position.x - Player.transform.position.
        if (distanceToNextCheckpoint >= 0)
            return;

        _checkpoints[_currentCheckpointIndex].PlayerLeftCheckpoint();
        _currentCheckpointIndex++;
        _checkpoints[_currentCheckpointIndex].PlayerHitCheckpoint();
    }

    public void KillPlayer()
    {
        StartCoroutine(KillPlayerCo());
    }

    private IEnumerator KillPlayerCo()
    {
        Player.Kill();
        Camera.IsFollowing = false;
        yield return new WaitForSeconds(2f);

        Camera.IsFollowing = true;

        if (_currentCheckpointIndex != -1)
                        _checkpoints[_currentCheckpointIndex].SpawnPlayer(Player);
    }
}
```

- CHECKPOINTS

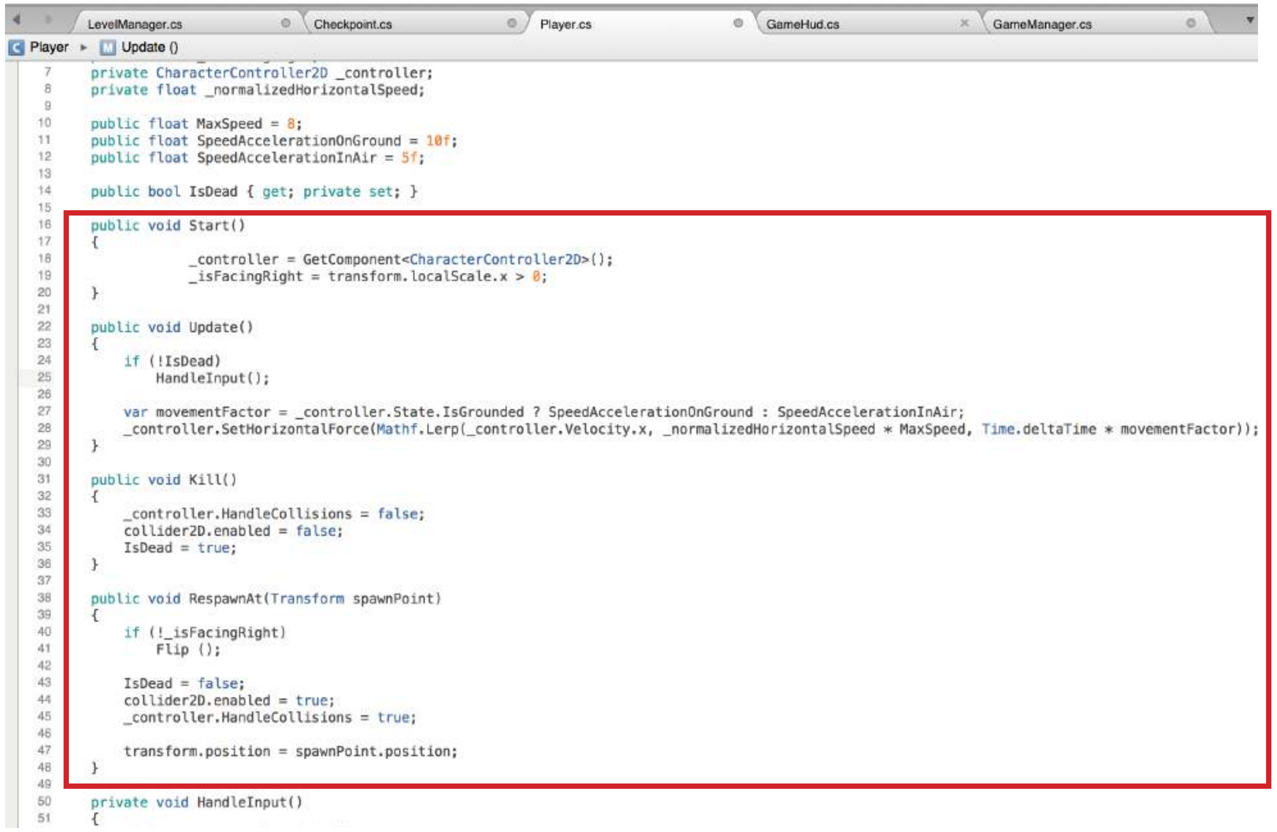
I think it would be silly not to implement checkpoints in my game, because really, every game has them! I started to outline my script with the classes that I would need for the checkpoints to work correctly.



```
using UnityEngine;
using System.Collections;

public class Checkpoint : MonoBehaviour
{
    public void Start()
    {

    }

    public void PlayerHitCheckpoint()
    {

    }

    private IEnumerator PlayerHitCheckpointCo()
    {
        yield break;
    }

    public void PlayerLeftCheckpoint()
    {

    }

    public void SpawnPlayer(Player Player)
    {
        Player.RespawnAt(transform);
    }

    public void AssignObjectToCheckpoint()
    {

    }


}
```

- CHECKPOINTS

Back in the "player.cs" script I need to add a few lines of code to tell Unity that when the character reached 0 health it needs to respawn at the spawn point, which is the last checkpoint the player had reached (if not back to the start).



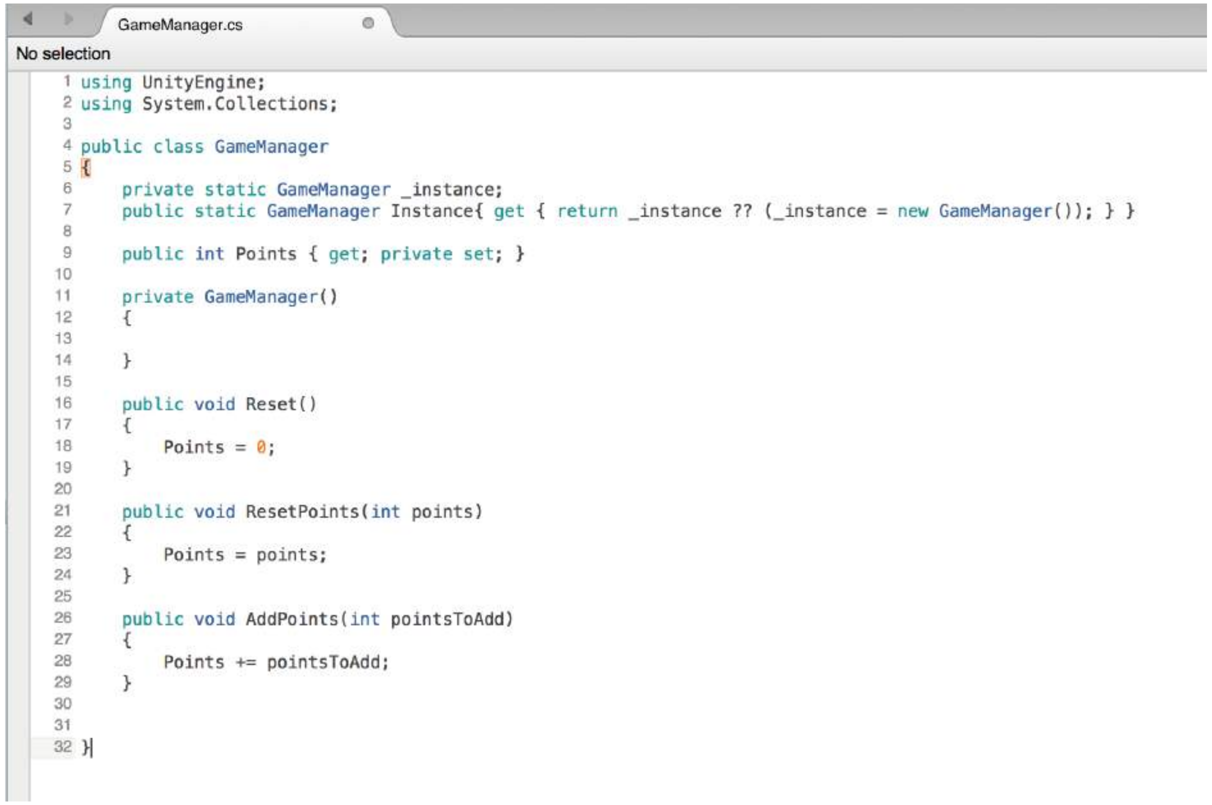
```
private CharacterController2D _controller;
private float _normalizedHorizontalSpeed;

public float MaxSpeed = 8;
public float SpeedAccelerationOnGround = 10f;
public float SpeedAccelerationInAir = 5f;

public bool IsDead { get; private set; }

public void Start()
{
            _controller = GetComponent<CharacterController2D>();
            _isFacingRight = transform.localScale.x > 0;
}

public void Update()
{
    if (!IsDead)
        HandleInput();

    var movementFactor = _controller.State.IsGrounded ? SpeedAccelerationOnGround : SpeedAccelerationInAir;
    _controller.SetHorizontalForce(Mathf.Lerp(_controller.Velocity.x, _normalizedHorizontalSpeed * MaxSpeed, Time.deltaTime * movementFactor));
}

public void Kill()
{
    _controller.HandleCollisions = false;
    collider2D.enabled = false;
    IsDead = true;
}

public void RespawnAt(Transform spawnPoint)
{
    if (!_isFacingRight)
        Flip ();

    IsDead = false;
    collider2D.enabled = true;
    _controller.HandleCollisions = true;

    transform.position = spawnPoint.position;
}

private void HandleInput()
{
```

GAME DEVELOPMENT

- HUD

Every game has some sort of Heads Up Display (HUD) so I thought it was right to include a simple one of my own.

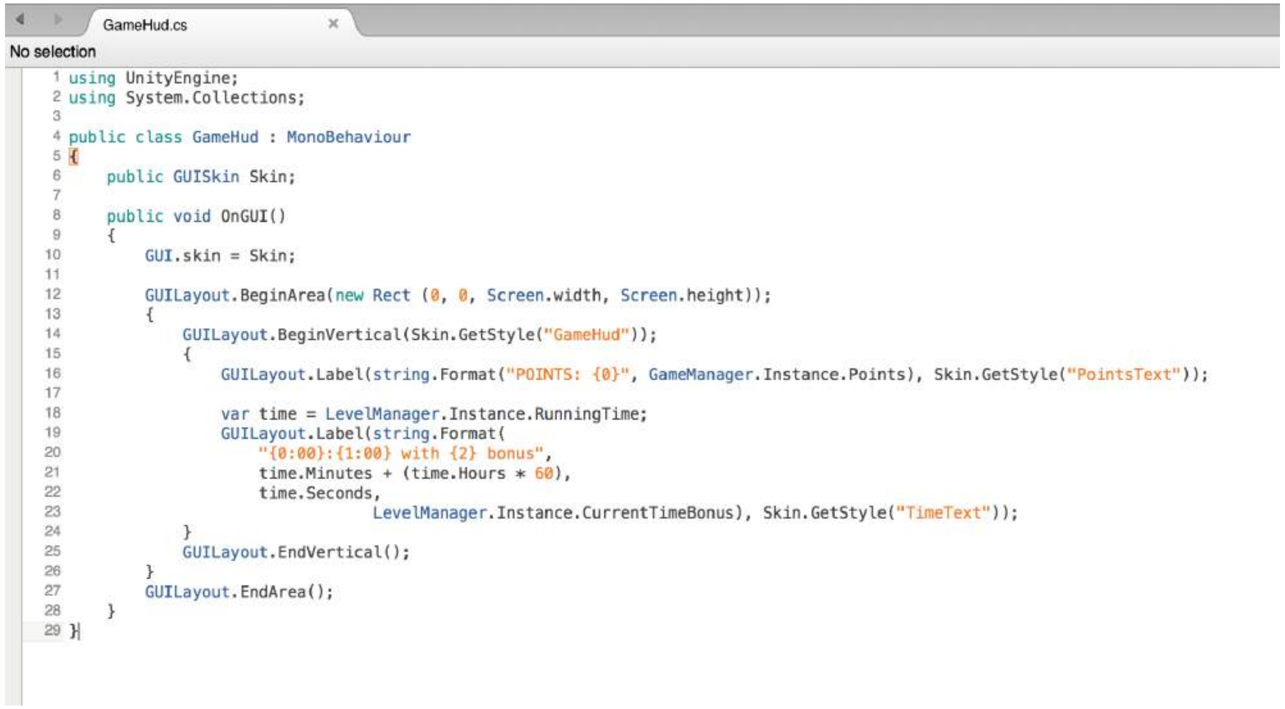
I want a points system in my game so there is a purpose to the game. The game manager script outlines the points system that I am going to implement.



```
using UnityEngine;
using System.Collections;

public class GameManager
{
    private static GameManager _instance;
    public static GameManager Instance{ get { return _instance ?? (_instance = new GameManager()); } }

    public int Points { get; private set; }

    private GameManager()
    {

    }

    public void Reset()
    {
        Points = 0;
    }

    public void ResetPoints(int points)
    {
        Points = points;
    }

    public void AddPoints(int pointsToAdd)
    {
        Points += pointsToAdd;
    }


}
```

- HUD

This is the main code for the game HUD; this displays the time in which the player is running. The style of the HUD will be edited in the inspector window in Unity.



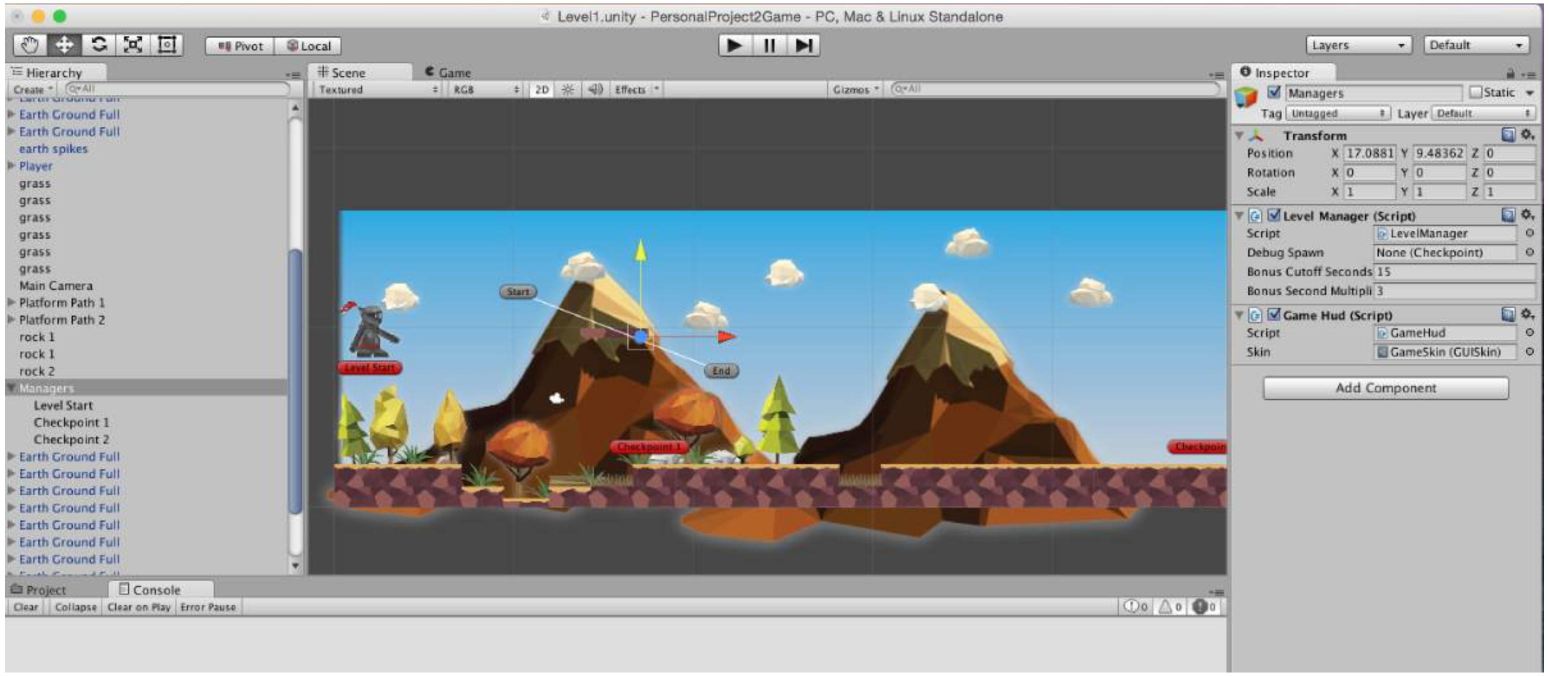
```
using UnityEngine;
using System.Collections;

public class GameHud : MonoBehaviour
{
    public GUISkin Skin;

    public void OnGUI()
    {
        GUI.skin = Skin;

        GUILayout.BeginArea(new Rect (0, 0, Screen.width, Screen.height));
        {
            GUILayout.BeginVertical(Skin.GetStyle("GameHud"));
            {
                GUILayout.Label(string.Format("POINTS: {0}", GameManager.Instance.Points), Skin.GetStyle("PointsText"));

                var time = LevelManager.Instance.RunningTime;
                GUILayout.Label(string.Format(
                    "{0:00}:{1:00} with {2} bonus",
                    time.Minutes + (time.Hours * 60),
                    time.Seconds,
                                LevelManager.Instance.CurrentTimeBonus), Skin.GetStyle("TimeText"));
            }
            GUILayout.EndVertical();
        }
        GUILayout.EndArea();
    }
}
```

GAME DEVELOPMENT

- HUD

This ties with the level manager. Back in Unity, as a test I make it so to get a bonus they must make it within 15 seconds to receive points, for a bonus, make it within 3.

These numbers will likely to change once a level is built. This text will then appear on the game HUD.



Managers ☐ Static
Tag Untagged Layer Default

Transform
Position X 17.0881 Y 9.48362 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

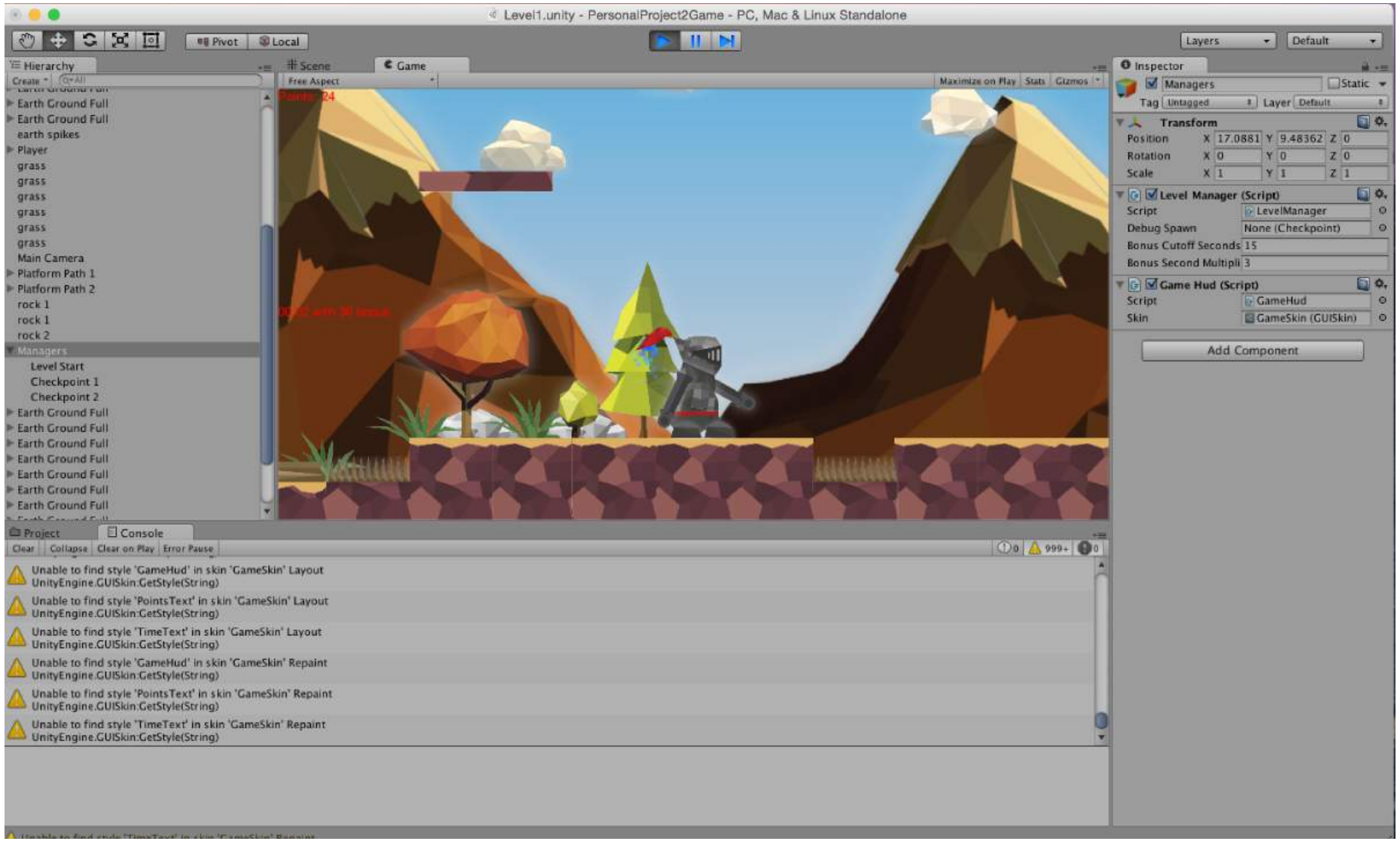
Level Manager (Script)
Script LevelManager
Debug Spawn None (Checkpoint)
Bonus Cutoff Seconds 15
Bonus Second Multipli 3

Game Hud (Script)
Script GameHud
Skin GameSkin (GUISkin)

GAME DEVELOPMENT

- HUD

When I hit play, you can see that a very basic HUD appears with the current time. Once a checkpoint is hit I get bonus points if I am in that specific time frame i set. It is all working correctly; it just looks a little ugly so next I will fix that.



Inspector
Managers — Static
Tag: Untagged — Layer: Default

Transform
Position X 17.0881 Y 9.48362 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

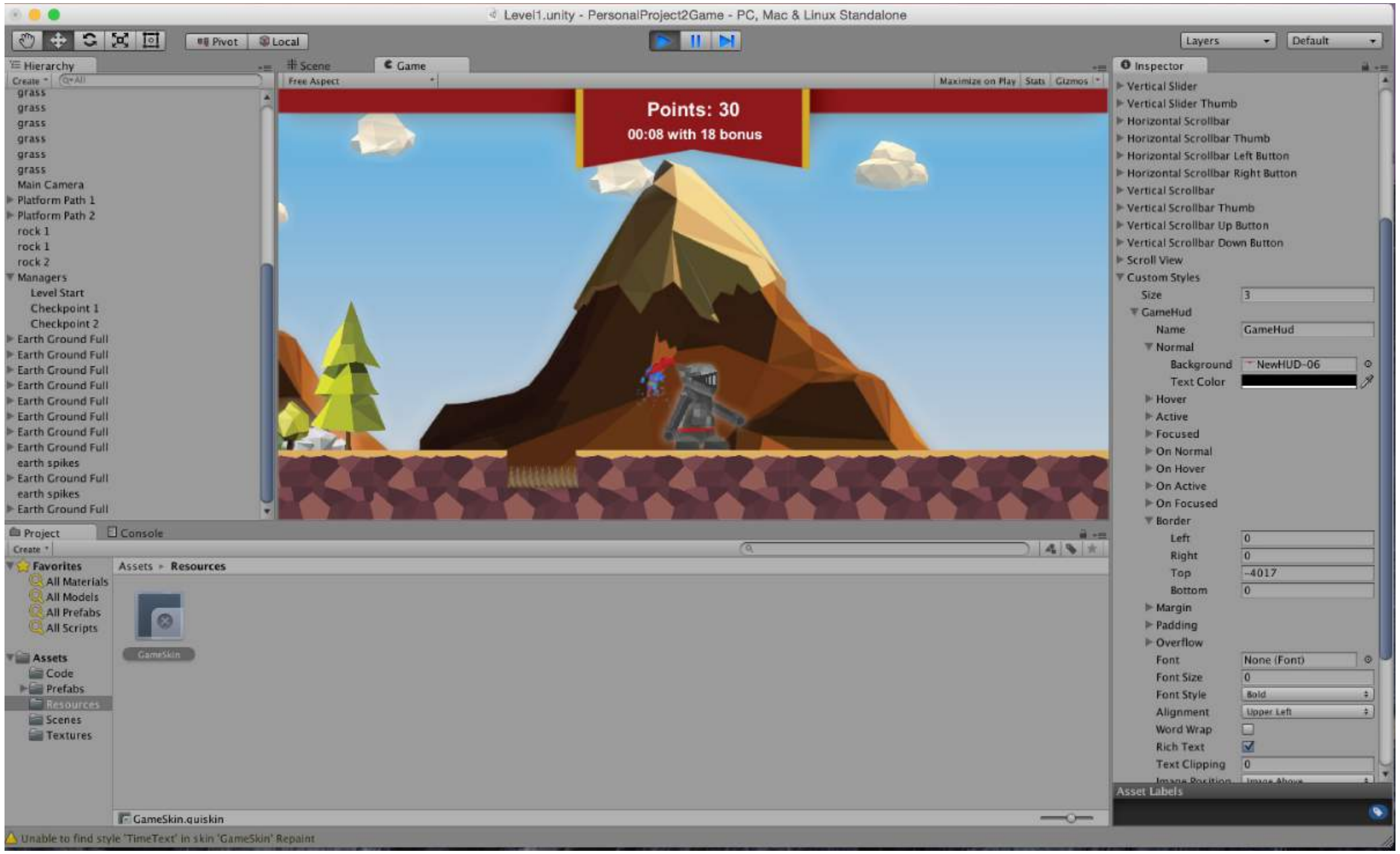
Level Manager (Script)
Script: LevelManager
Debug Spawn: None (Checkpoint)
Bonus Cutoff Seconds: 15
Bonus Second Multipli: 3

Game Hud (Script)
Script: GameHud
Skin: GameSkin (GUISkin)

```
Unable to find style 'GameHud' in skin 'GameSkin' Layout
UnityEngine.GUISkin:GetStyle(String)
Unable to find style 'PointsText' in skin 'GameSkin' Layout
UnityEngine.GUISkin:GetStyle(String)
Unable to find style 'TimeText' in skin 'GameSkin' Layout
UnityEngine.GUISkin:GetStyle(String)
Unable to find style 'GameHud' in skin 'GameSkin' Repaint
UnityEngine.GUISkin:GetStyle(String)
Unable to find style 'PointsText' in skin 'GameSkin' Repaint
UnityEngine.GUISkin:GetStyle(String)
Unable to find style 'TimeText' in skin 'GameSkin' Repaint
UnityEngine.GUISkin:GetStyle(String)
```

- HUD

With a bit of tweaking in the game skin I created, have kept my HUD in the same Medieval style and it looks a lot simpler and easier to understand now with a banner I designed in Photoshop.



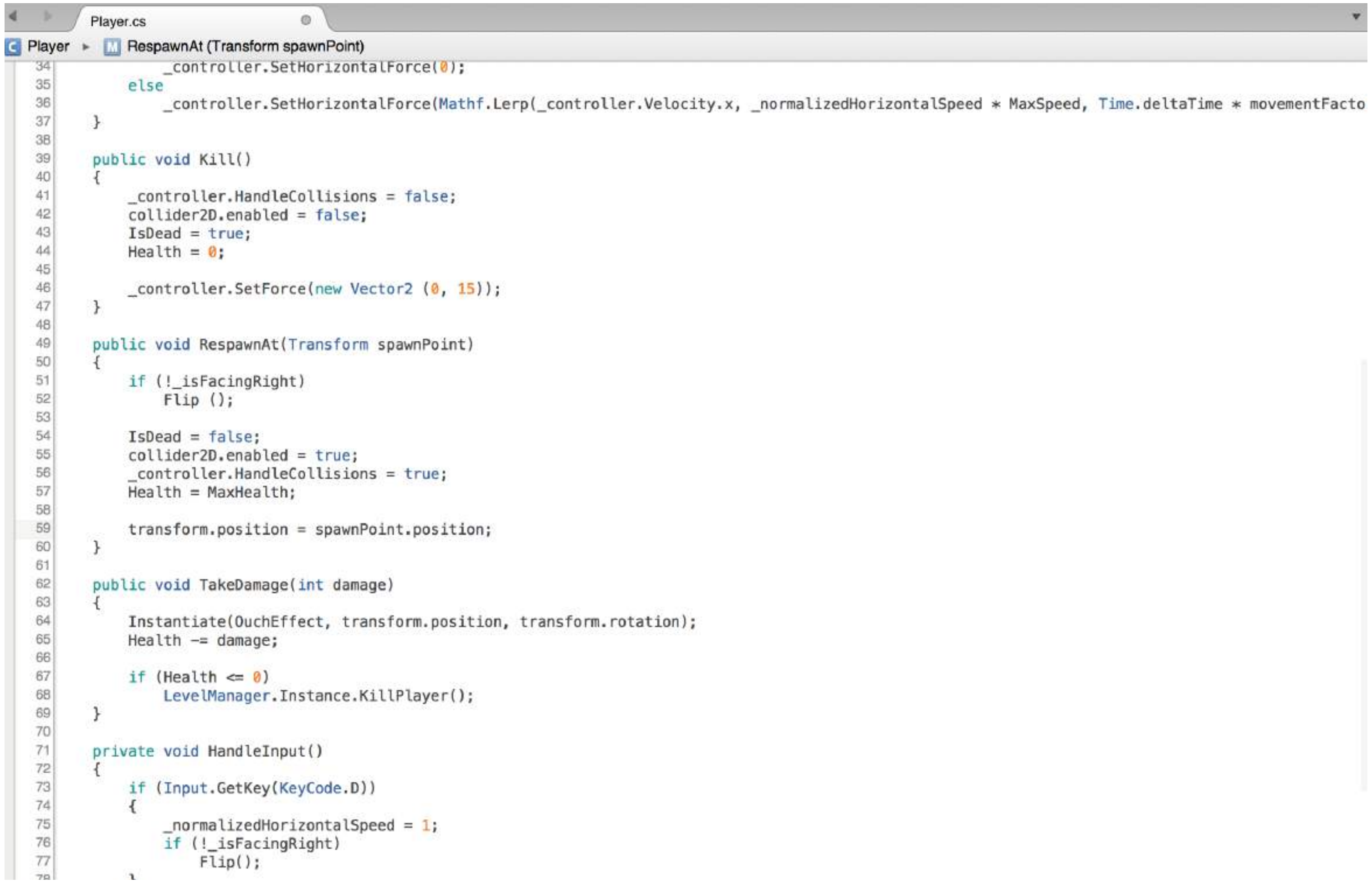
Size 3
GameHud
Name GameHud
Normal
Background NewHUD-06
Text Color
Border
Left 0
Right 0
Top -4017
Bottom 0
Font None (Font)
Font Size 0
Font Style Bold
Alignment Upper Left
Word Wrap ☐
Rich Text ✓
Text Clipping 0
Image Position Image Above

Unable to find style 'TimeText' in skin 'GameSkin' Repaint

- HEALTH & DAMAGE

I implemented health and damage code on player script.

What this does is basically when something hits player, like a cannon ball; we get damaged and the knight's health bar goes down from its max health (100).


Player.cs

Player ▸ RespawnAt (Transform spawnPoint)

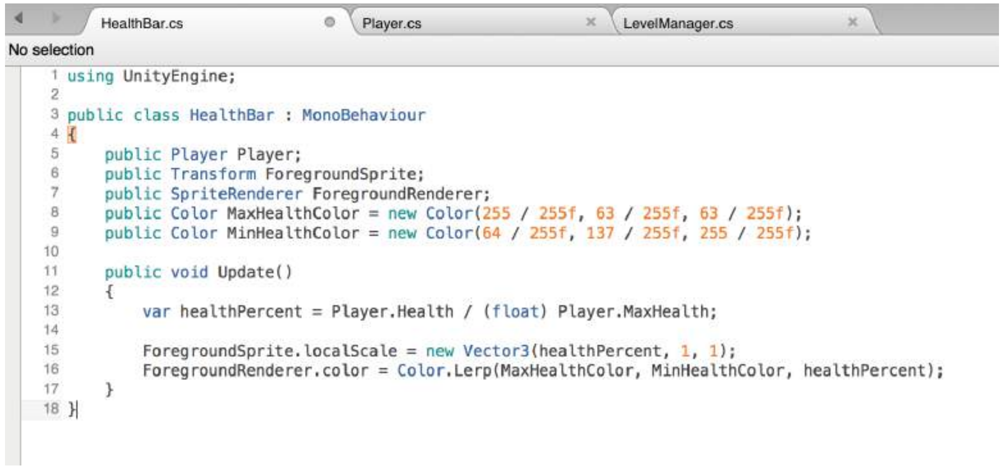
```
                _controller.SetHorizontalForce(0);
            else
                _controller.SetHorizontalForce(Mathf.Lerp(_controller.Velocity.x, _normalizedHorizontalSpeed * MaxSpeed, Time.deltaTime * movementFacto
        }

        public void Kill()
        {
            _controller.HandleCollisions = false;
            collider2D.enabled = false;
            IsDead = true;
            Health = 0;

            _controller.SetForce(new Vector2 (0, 15));
        }

        public void RespawnAt(Transform spawnPoint)
        {
            if (!_isFacingRight)
                Flip ();

            IsDead = false;
            collider2D.enabled = true;
            _controller.HandleCollisions = true;
            Health = MaxHealth;

            transform.position = spawnPoint.position;
        }

        public void TakeDamage(int damage)
        {
            Instantiate(OuchEffect, transform.position, transform.rotation);
            Health -= damage;

            if (Health <= 0)
                LevelManager.Instance.KillPlayer();
        }

        private void HandleInput()
        {
            if (Input.GetKey(KeyCode.D))
            {
                _normalizedHorizontalSpeed = 1;
                if (!_isFacingRight)
                    Flip();
```

GAME DEVELOPMENT

- HEALTH & DAMAGE

I want a health bar to appear above the player when the user is moving the knight around the world.

I have designed the Photoshop file for the graphic side but first I need to write the health bar script that will visualise the player health percentage out of 100.



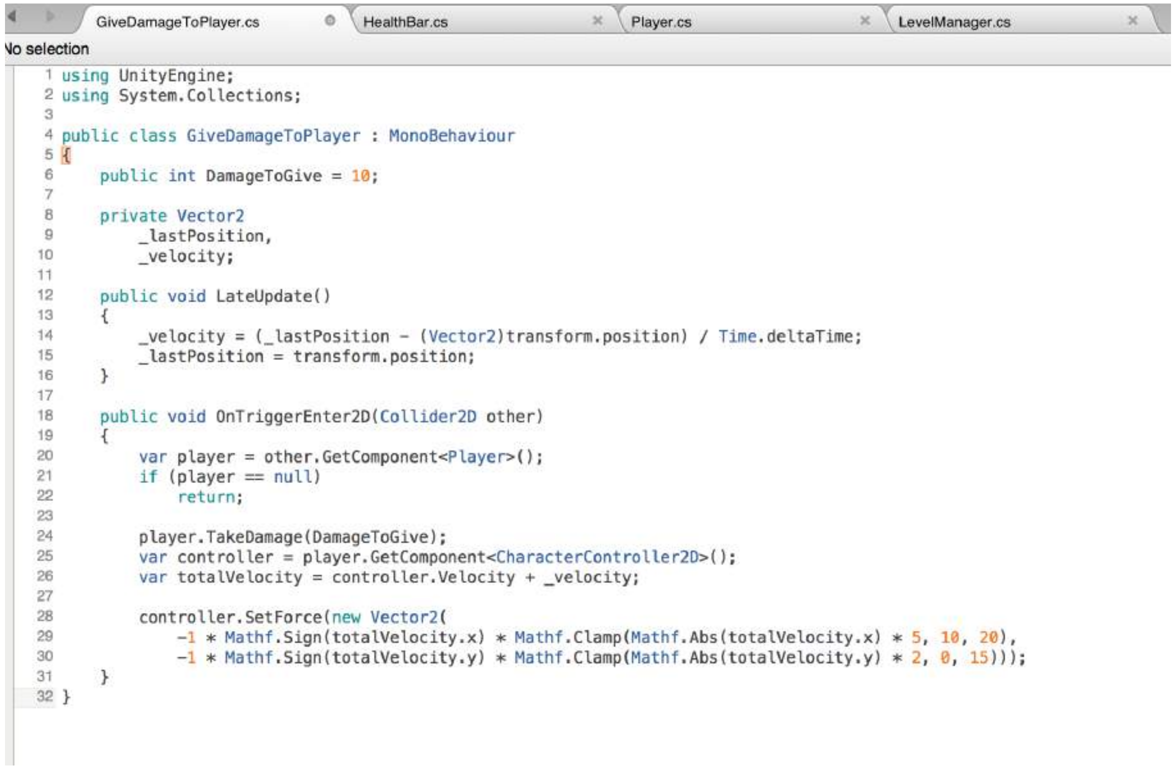
```
using UnityEngine;

public class HealthBar : MonoBehaviour
{
    public Player Player;
    public Transform ForegroundSprite;
    public SpriteRenderer ForegroundRenderer;
    public Color MaxHealthColor = new Color(255 / 255f, 63 / 255f, 63 / 255f);
    public Color MinHealthColor = new Color(64 / 255f, 137 / 255f, 255 / 255f);

    public void Update()
    {
        var healthPercent = Player.Health / (float) Player.MaxHealth;

        ForegroundSprite.localScale = new Vector3(healthPercent, 1, 1);
        ForegroundRenderer.color = Color.Lerp(MaxHealthColor, MinHealthColor, healthPercent);
    }
}
```

GAME DEVELOPMENT

- HEALTH & DAMAGE

I create a new script that will give damage to the player and take a certain amount of damage off them if they get hit.

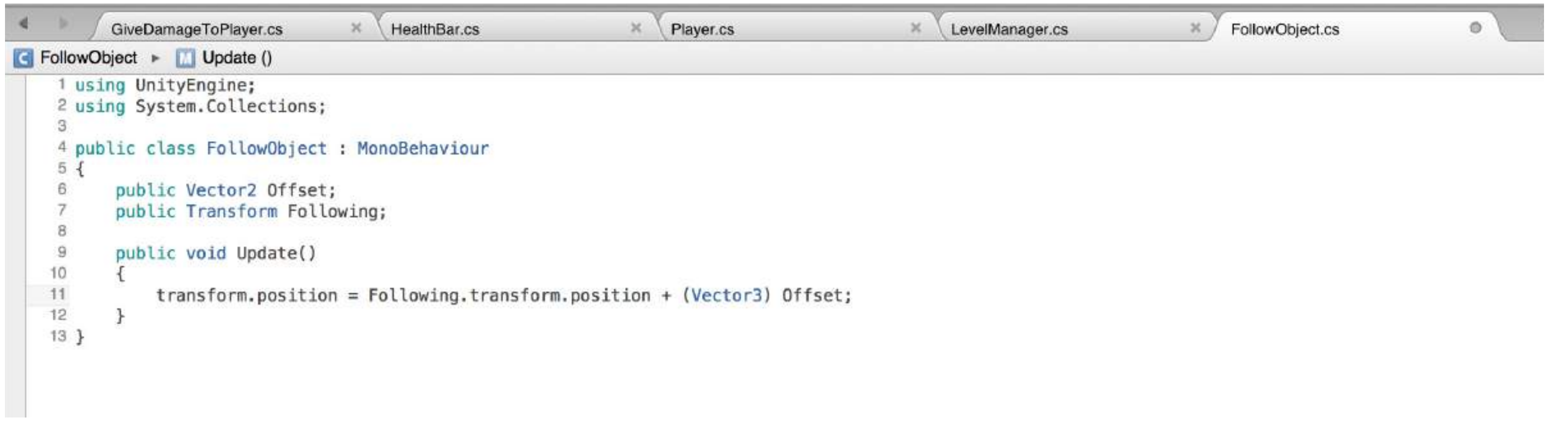
I wont be able to test this yet until my health bar is fully functioning so I can see visually the health depleting.



```
using UnityEngine;
using System.Collections;

public class GiveDamageToPlayer : MonoBehaviour
{
    public int DamageToGive = 10;

    private Vector2
        _lastPosition,
        _velocity;

    public void LateUpdate()
    {
        _velocity = (_lastPosition - (Vector2)transform.position) / Time.deltaTime;
        _lastPosition = transform.position;
    }

    public void OnTriggerEnter2D(Collider2D other)
    {
        var player = other.GetComponent<Player>();
        if (player == null)
            return;

        player.TakeDamage(DamageToGive);
        var controller = player.GetComponent<CharacterController2D>();
        var totalVelocity = controller.Velocity + _velocity;

        controller.SetForce(new Vector2(
            -1 * Mathf.Sign(totalVelocity.x) * Mathf.Clamp(Mathf.Abs(totalVelocity.x) * 5, 10, 20),
            -1 * Mathf.Sign(totalVelocity.y) * Mathf.Clamp(Mathf.Abs(totalVelocity.y) * 2, 0, 15)));
    }
}
```

- HEALTH & DAMAGE

As I have said, I want the health bar to follow the player around the world, i created a new script that will follow an object, this object will be the player object when I assign it in the inspector window in Unity.



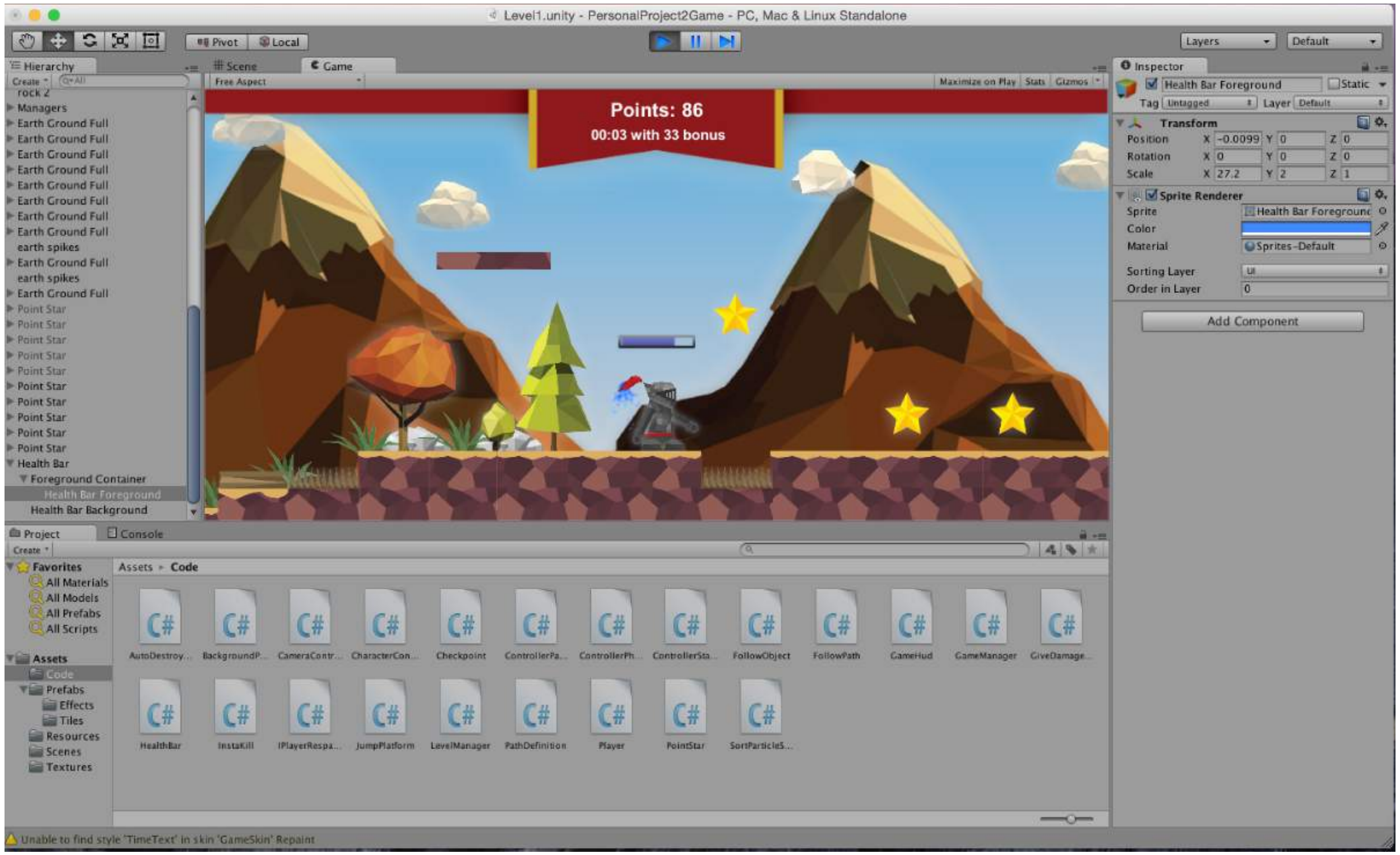
```
using UnityEngine;
using System.Collections;

public class FollowObject : MonoBehaviour
{
    public Vector2 Offset;
    public Transform Following;

    public void Update()
    {
        transform.position = Following.transform.position + (Vector3) Offset;
    }
}
```

GAME DEVELOPMENT

- HEALTH & DAMAGE

Back in Unity, the health bar is working great and as it should.

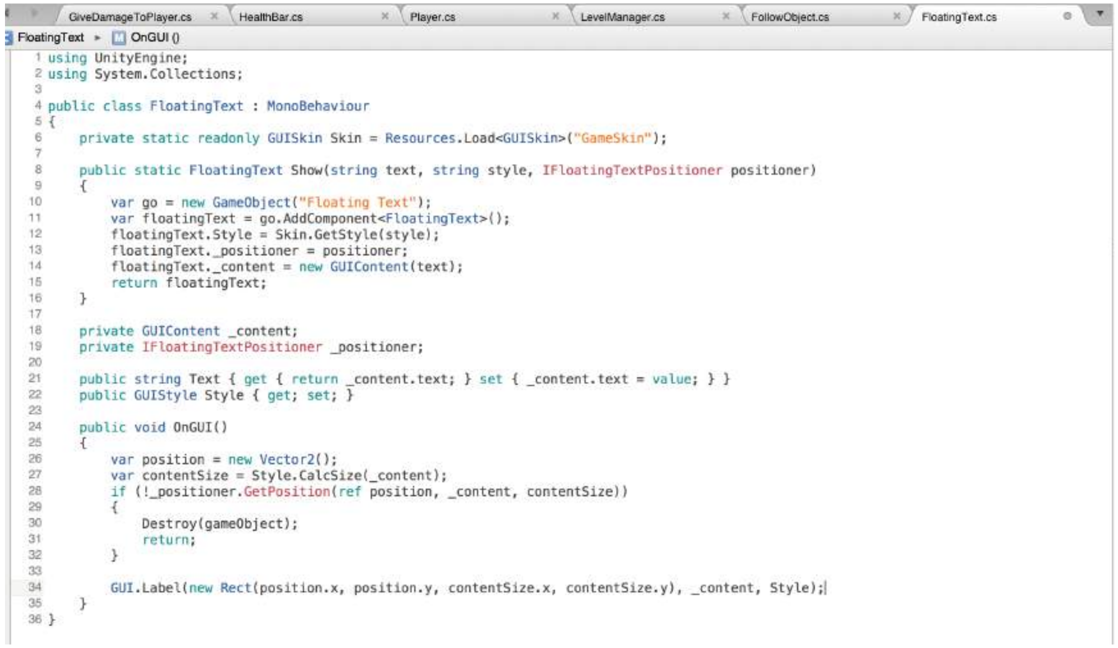
Once I assigned all the components to the health bar the visual indicator works to great effect and now it is looking a little more like a completed 2D character design.



GAME DEVELOPMENT

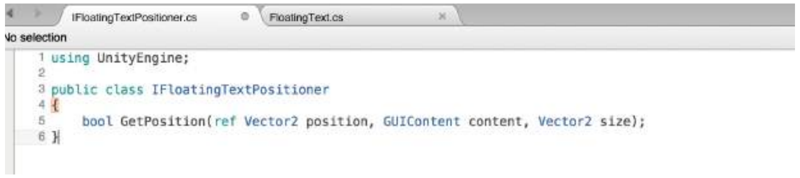
- FLOATING TEXT

I want floating text to appear when a star or health pack is collected by the player to give the user a clearer indication of what that collectable does. This script is how it is going to be used, at the moment I have not implemented it yet, just telling unity how it will be used.



```
using UnityEngine;
using System.Collections;

public class FloatingText : MonoBehaviour
{
    private static readonly GUISkin Skin = Resources.Load<GUISkin>("GameSkin");

    public static FloatingText Show(string text, string style, IFloatingTextPositioner positioner)
    {
        var go = new GameObject("Floating Text");
        var floatingText = go.AddComponent<FloatingText>();
        floatingText.Style = Skin.GetStyle(style);
        floatingText._positioner = positioner;
        floatingText._content = new GUIContent(text);
        return floatingText;
    }

    private GUIContent _content;
    private IFloatingTextPositioner _positioner;

    public string Text { get { return _content.text; } set { _content.text = value; } }
    public GUIStyle Style { get; set; }

    public void OnGUI()
    {
        var position = new Vector2();
        var contentSize = Style.CalcSize(_content);
        if (!_positioner.GetPosition(ref position, _content, contentSize))
        {
            Destroy(gameObject);
            return;
        }

        GUI.Label(new Rect(position.x, position.y, contentSize.x, contentSize.y), _content, Style);
    }
}
```

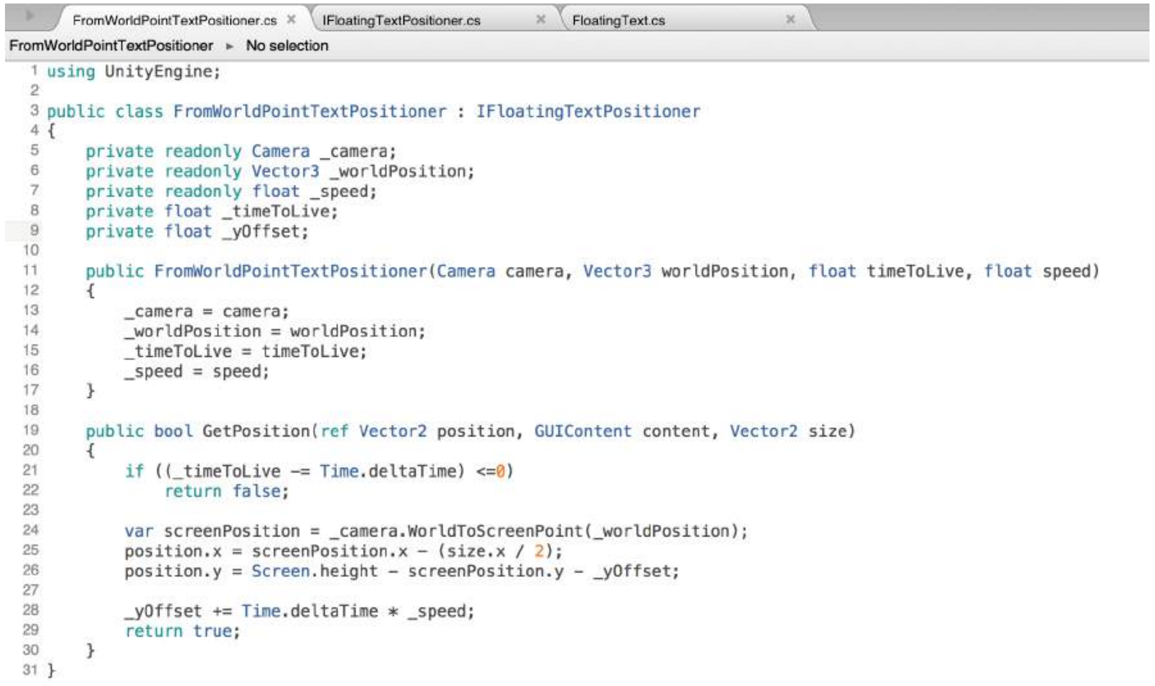
This second bit of code for floating text position, compiles with the code I have just written, so the floating text positioner class works.



```
using UnityEngine;

public class IFloatingTextPositioner
{
    bool GetPosition(ref Vector2 position, GUIContent content, Vector2 size);
}
```

- FLOATING TEXT

In this script the Y offset is scaled by the speed that we want to go in. This will link to "PointStar.cs" script. Basically, these scripts work together so when a collectable is obtained, text will rise up and float above the player with a visual indicator of how much it is worth.

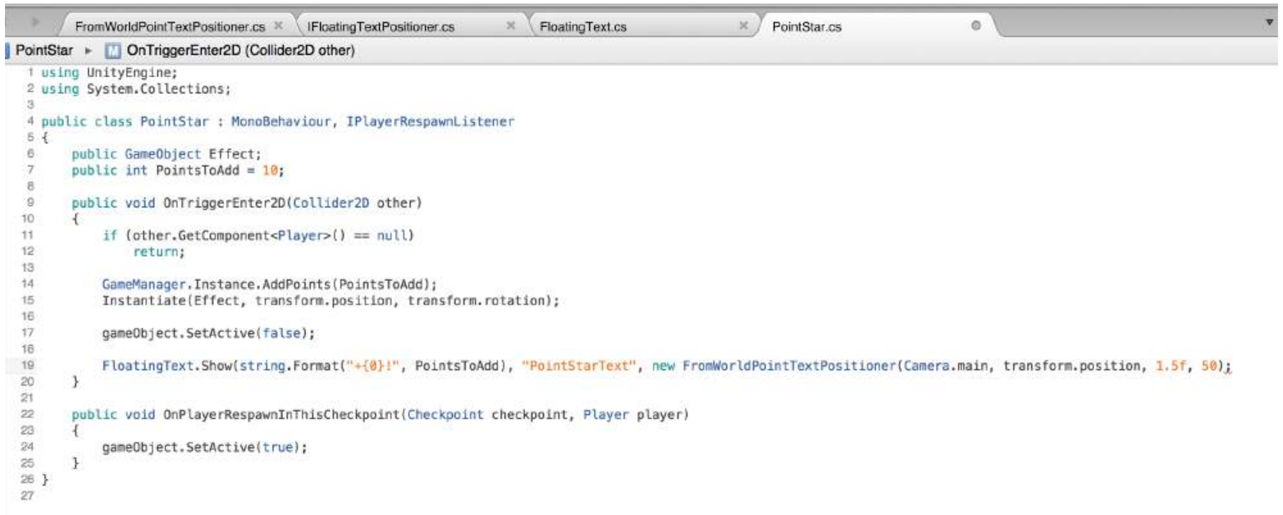

FromWorldPointTextPositioner.cs
IFloatingTextPositioner.cs
FloatingText.cs
FromWorldPointTextPositioner ▸ No selection

```
using UnityEngine;

public class FromWorldPointTextPositioner : IFloatingTextPositioner
{
    private readonly Camera _camera;
    private readonly Vector3 _worldPosition;
    private readonly float _speed;
    private float _timeToLive;
    private float _yOffset;

    public FromWorldPointTextPositioner(Camera camera, Vector3 worldPosition, float timeToLive, float speed)
    {
        _camera = camera;
        _worldPosition = worldPosition;
        _timeToLive = timeToLive;
        _speed = speed;
    }

    public bool GetPosition(ref Vector2 position, GUIContent content, Vector2 size)
    {
        if ((_timeToLive -= Time.deltaTime) <=0)
            return false;

        var screenPosition = _camera.WorldToScreenPoint(_worldPosition);
        position.x = screenPosition.x - (size.x / 2);
        position.y = Screen.height - screenPosition.y - _yOffset;

        _yOffset += Time.deltaTime * _speed;
        return true;
    }
}
```

GAME DEVELOPMENT

- FLOATING TEXT

Now that I have sorted the gravitational values of the floating text, this new line of code works so that it will show "+10!" when a star is collected.

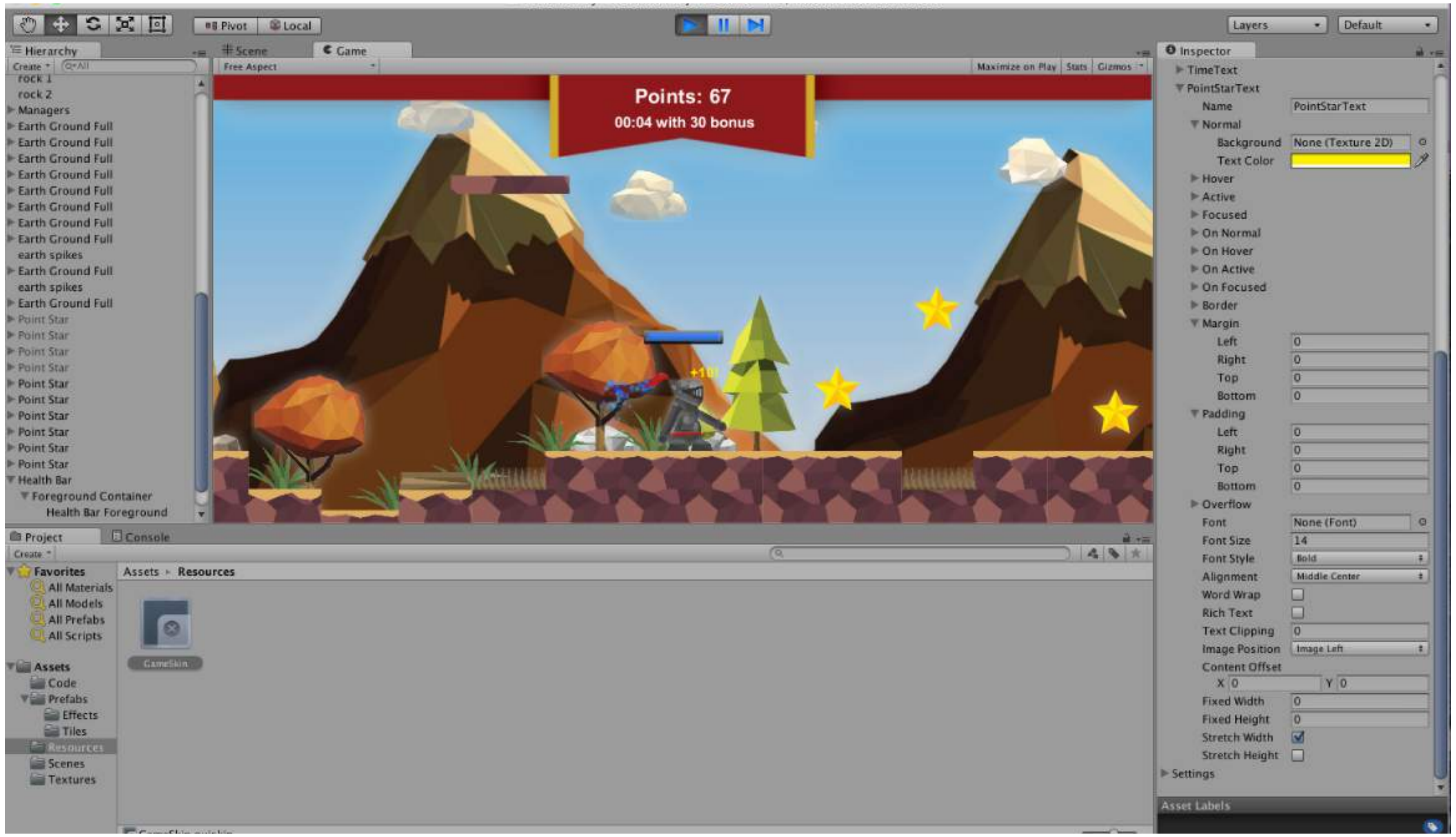


FromWorldPointTextPositioner.cs
IFloatingTextPositioner.cs
FloatingText.cs
PointStar.cs
PointStar ▸ OnTriggerEnter2D (Collider2D other)

```
using UnityEngine;
using System.Collections;

public class PointStar : MonoBehaviour, IPlayerRespawnListener
{
    public GameObject Effect;
    public int PointsToAdd = 10;

    public void OnTriggerEnter2D(Collider2D other)
    {
        if (other.GetComponent<Player>() == null)
            return;

        GameManager.Instance.AddPoints(PointsToAdd);
        Instantiate(Effect, transform.position, transform.rotation);

        gameObject.SetActive(false);

        FloatingText.Show(string.Format("+{0}!", PointsToAdd), "PointStarText", new FromWorldPointTextPositioner(Camera.main, transform.position, 1.5f, 50));
    }

    public void OnPlayerRespawnInThisCheckpoint(Checkpoint checkpoint, Player player)
    {
        gameObject.SetActive(true);
    }
}
```

- FLOATING TEXT

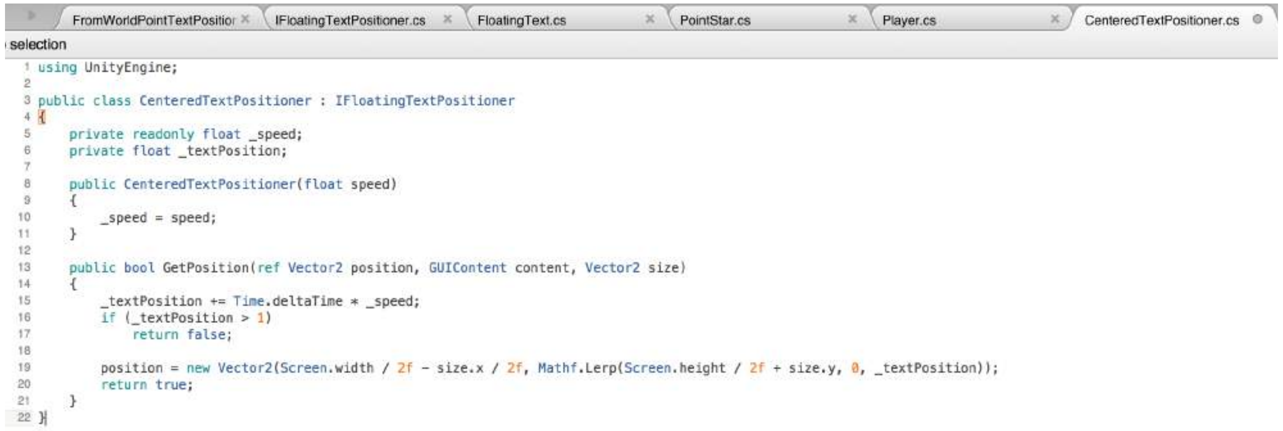
When I go back into Unity and press play, the floating text is working, a "+10!" floats vertically as a star is collected which is what I was looking for, it does give a bit more of a response to the player when the pick up a star, also I changed the colour of the text to yellow, to match the star colour.



Inspector
- TimeText
- PointStarText
  - Name: PointStarText
  - Normal
    - Background: None (Texture 2D)
    - Text Color: 🟨
  - Hover
  - Active
  - Focused
  - On Normal
  - On Hover
  - On Active
  - On Focused
  - Border
  - Margin
    - Left: 0
    - Right: 0
    - Top: 0
    - Bottom: 0
  - Padding
    - Left: 0
    - Right: 0
    - Top: 0
    - Bottom: 0
  - Overflow
  - Font: None (Font)
  - Font Size: 14
  - Font Style: Bold
  - Alignment: Middle Center
  - Word Wrap: ☐
  - Rich Text: ☐
  - Text Clipping: 0
  - Image Position: Image Left
  - Content Offset: X 0 Y 0
  - Fixed Width: 0
  - Fixed Height: 0
  - Stretch Width: ✓
  - Stretch Height: ☐
- Settings

- FLOATING TEXT

While I am adding floating text, I have completed the star system, now what I want to do is add code for checkpoint class, when a checkpoint is passed, text will appear to notify the player. As I have seen, this will give a bit more of a response to the player when passing a checkpoint because at the moment they don't know where the checkpoints are, they only know when they die.

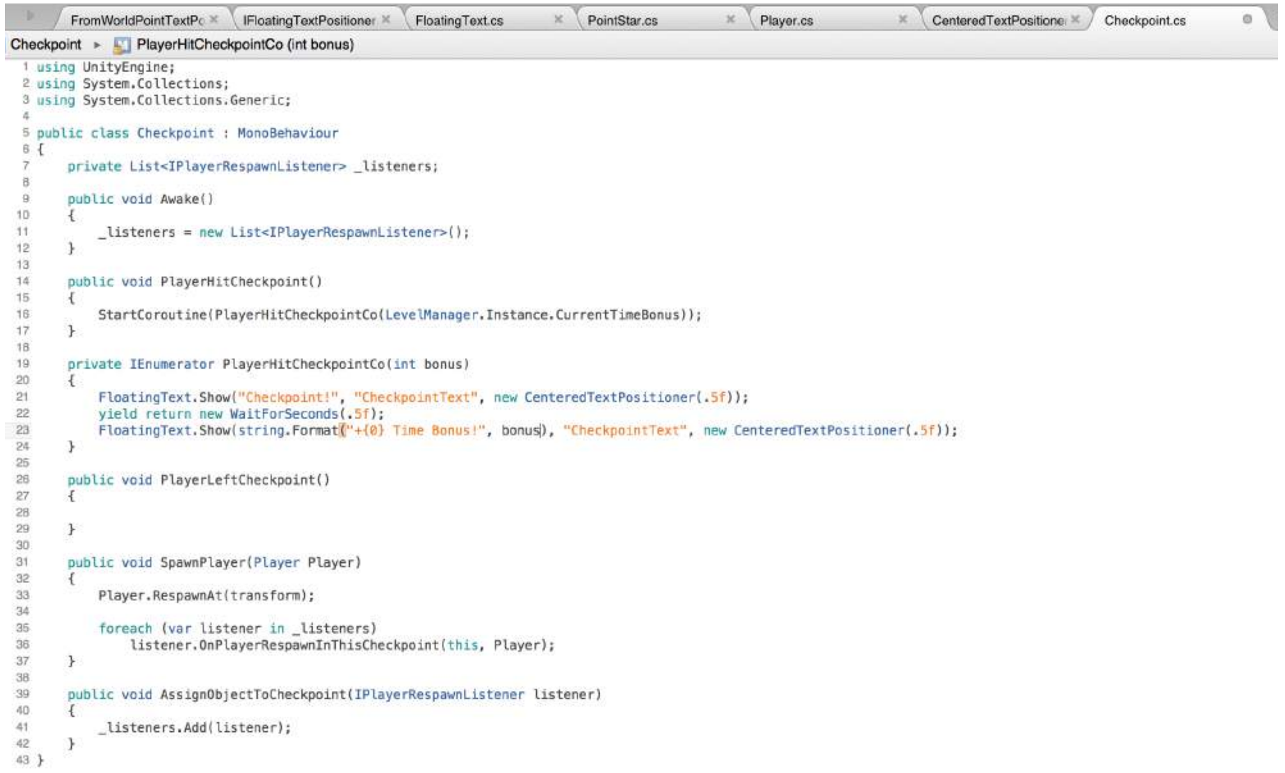

FromWorldPointTextPositior
IFloatingTextPositioner.cs
FloatingText.cs
PointStar.cs
Player.cs
CenteredTextPositioner.cs
selection

```
using UnityEngine;

public class CenteredTextPositioner : IFloatingTextPositioner
{
    private readonly float _speed;
    private float _textPosition;

    public CenteredTextPositioner(float speed)
    {
        _speed = speed;
    }

    public bool GetPosition(ref Vector2 position, GUIContent content, Vector2 size)
    {
        _textPosition += Time.deltaTime * _speed;
        if (_textPosition > 1)
            return false;

        position = new Vector2(Screen.width / 2f - size.x / 2f, Mathf.Lerp(Screen.height / 2f + size.y, 0, _textPosition));
        return true;
    }
}
```

GAME DEVELOPMENT

- FLOATING TEXT

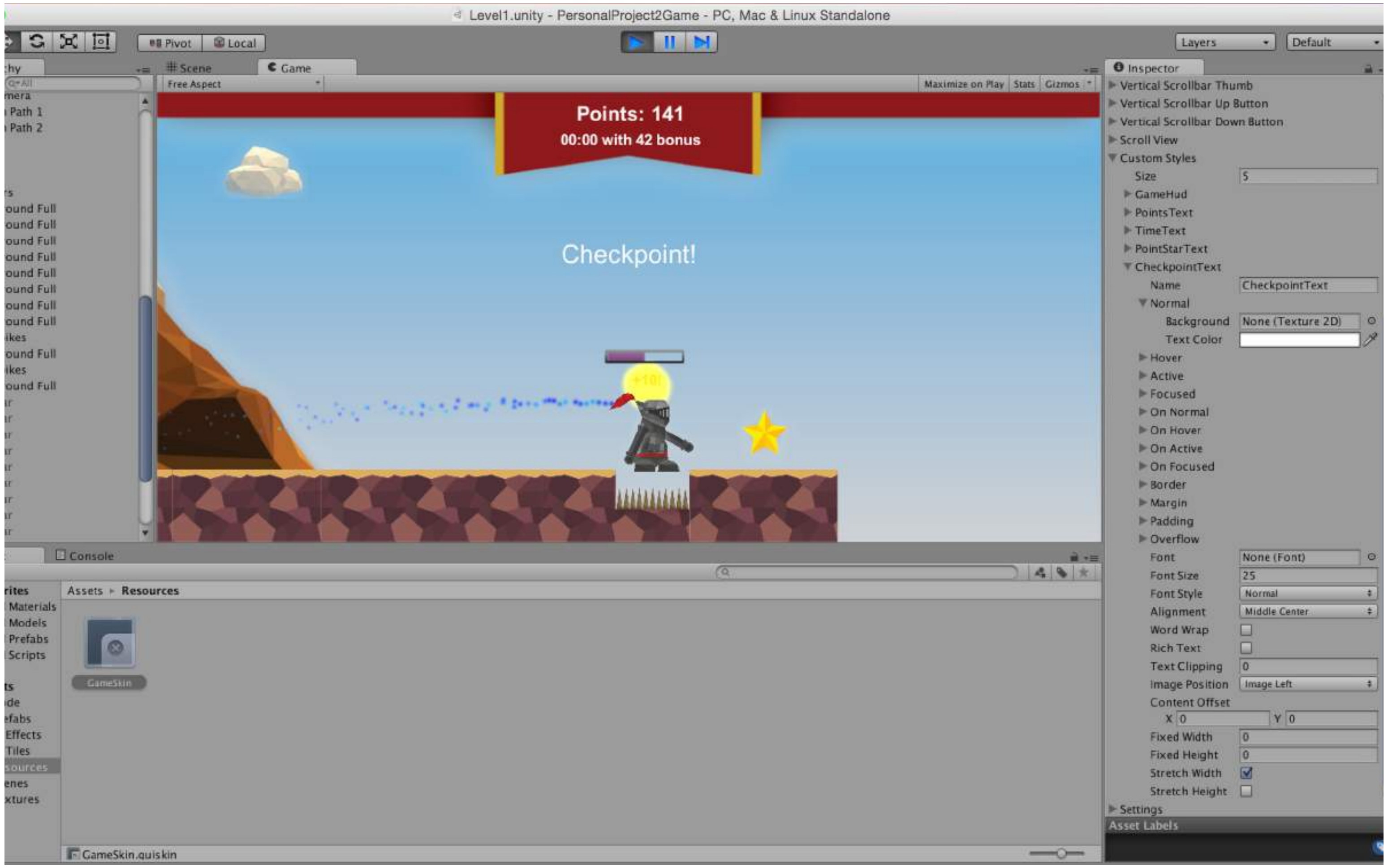
Once a checkpoint has been passed the time bonus text will appear and float just like the text for the star. But instead of displaying a numerical value, the text will say "Checkpoint!" and then a time bonus of how many seconds will appear for a brief time on screen.



```csharp
using UnityEngine;
using System.Collections;
using System.Collections.Generic;

public class Checkpoint : MonoBehaviour
{
    private List<IPlayerRespawnListener> _listeners;

    public void Awake()
    {
        _listeners = new List<IPlayerRespawnListener>();
    }

    public void PlayerHitCheckpoint()
    {
        StartCoroutine(PlayerHitCheckpointCo(LevelManager.Instance.CurrentTimeBonus));
    }

    private IEnumerator PlayerHitCheckpointCo(int bonus)
    {
        FloatingText.Show("Checkpoint!", "CheckpointText", new CenteredTextPositioner(.5f));
        yield return new WaitForSeconds(.5f);
        FloatingText.Show(string.Format("+{0} Time Bonus!", bonus), "CheckpointText", new CenteredTextPositioner(.5f));
    }

    public void PlayerLeftCheckpoint()
    {

    }

    public void SpawnPlayer(Player Player)
    {
        Player.RespawnAt(transform);

        foreach (var listener in _listeners)
            listener.OnPlayerRespawnInThisCheckpoint(this, Player);
    }

    public void AssignObjectToCheckpoint(IPlayerRespawnListener listener)
    {
        _listeners.Add(listener);
    }
}
```

- FLOATING TEXT

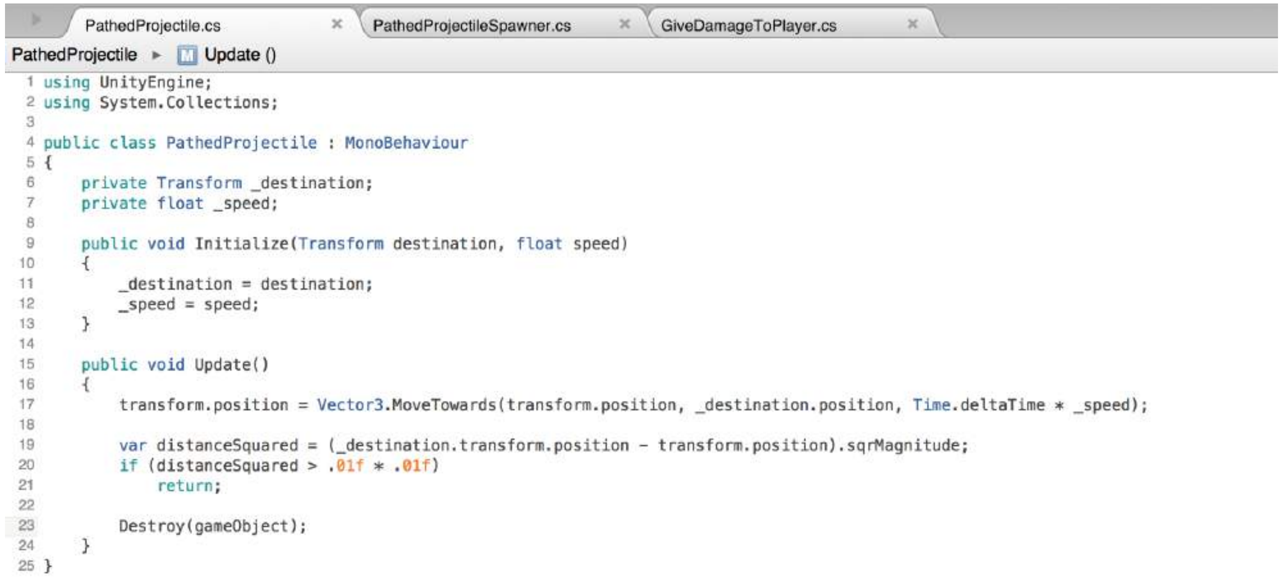
When I tested this code out the checkpoint text appears when a checkpoint is passed by the player and floats vertically, this is what I want and will serve its purpose nicely for my game. Note I have edited the text slightly, I made it larger and white, as this has a little bit more importance to the player when in the game.



```
▶ Vertical Scrollbar Thumb
▶ Vertical Scrollbar Up Button
▶ Vertical Scrollbar Down Button
▶ Scroll View
▼ Custom Styles
    Size                5
  ▶ GameHud
  ▶ PointsText
  ▶ TimeText
  ▶ PointStarText
  ▼ CheckpointText
      Name              CheckpointText
    ▼ Normal
        Background      None (Texture 2D)
        Text Color
    ▶ Hover
    ▶ Active
    ▶ Focused
    ▶ On Normal
    ▶ On Hover
    ▶ On Active
    ▶ On Focused
    ▶ Border
    ▶ Margin
    ▶ Padding
    ▶ Overflow
      Font              None (Font)
      Font Size         25
      Font Style        Normal
      Alignment         Middle Center
      Word Wrap         ☐
      Rich Text         ☐
      Text Clipping     0
      Image Position    Image Left
      Content Offset
        X 0             Y 0
      Fixed Width       0
      Fixed Height      0
      Stretch Width     ☑
      Stretch Height    ☐
▶ Settings
```

- PATHED PROJECTILES

My game will have cannons that fire cannon balls that the player will need to dodge or destroy. This pathed projectile code states where the cannon ball will end up basically calculated by the distance and speed. This path will be a straight line, hence the name "Pathed Projectile".



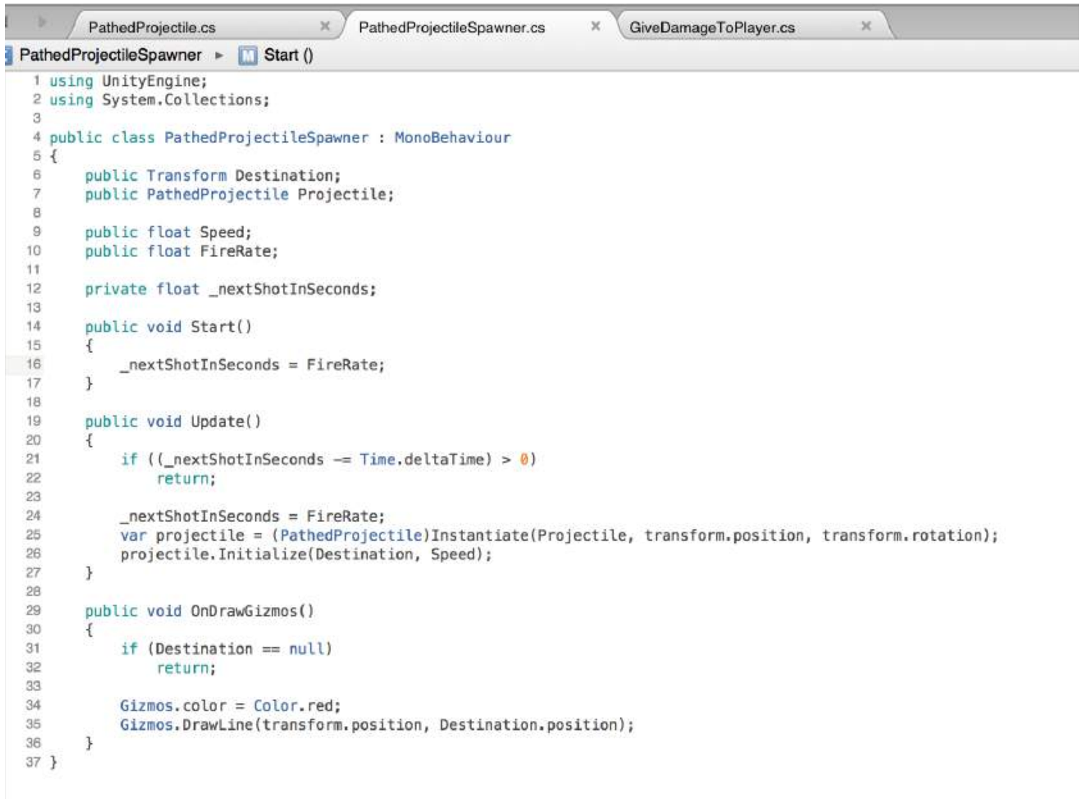
```
using UnityEngine;
using System.Collections;

public class PathedProjectile : MonoBehaviour
{
    private Transform _destination;
    private float _speed;

    public void Initialize(Transform destination, float speed)
    {
        _destination = destination;
        _speed = speed;
    }

    public void Update()
    {
        transform.position = Vector3.MoveTowards(transform.position, _destination.position, Time.deltaTime * _speed);

        var distanceSquared = (_destination.transform.position - transform.position).sqrMagnitude;
        if (distanceSquared > .01f * .01f)
            return;

        Destroy(gameObject);
    }
}
```

GAME DEVELOPMENT

- PATHED PROJECTILES

I have written the code for where the cannon ball is going to go but now I need to define where it will spawn. This code will allow the spawner to be active for the cannon ball. Then it will determine how fast it is going to spawn and what fire rate it shoots a ball, so for example every 1 second.

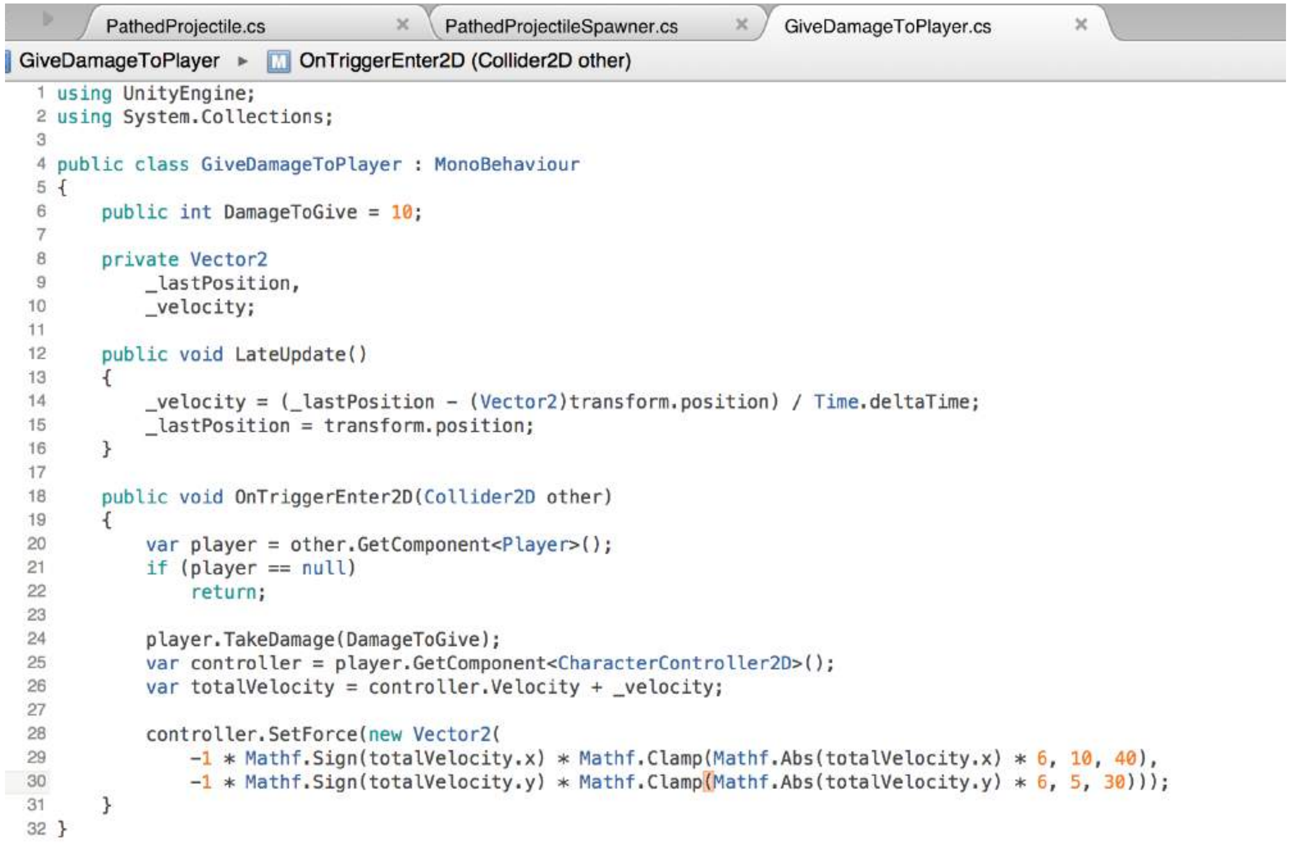
I have also implemented a line, similar to the moving platforms line so I can see the path in the scene view clearer, its colour is red.


PathedProjectile.cs
PathedProjectileSpawner.cs
GiveDamageToPlayer.cs
PathedProjectileSpawner ▸ Start ()

```
using UnityEngine;
using System.Collections;

public class PathedProjectileSpawner : MonoBehaviour
{
    public Transform Destination;
    public PathedProjectile Projectile;

    public float Speed;
    public float FireRate;

    private float _nextShotInSeconds;

    public void Start()
    {
        _nextShotInSeconds = FireRate;
    }

    public void Update()
    {
        if ((_nextShotInSeconds -= Time.deltaTime) > 0)
            return;

        _nextShotInSeconds = FireRate;
        var projectile = (PathedProjectile)Instantiate(Projectile, transform.position, transform.rotation);
        projectile.Initialize(Destination, Speed);
    }

    public void OnDrawGizmos()
    {
        if (Destination == null)
            return;

        Gizmos.color = Color.red;
        Gizmos.DrawLine(transform.position, Destination.position);
    }
}
```

- PATHED PROJECTILES

I want the cannon ball to give damage to the player if it hits them. This code will do just that, In this case, the ball will take 10 damage off the player if hit. Note that the player has 100 health, so 10 hits by this will kill the player, which is the game mechanic I want.

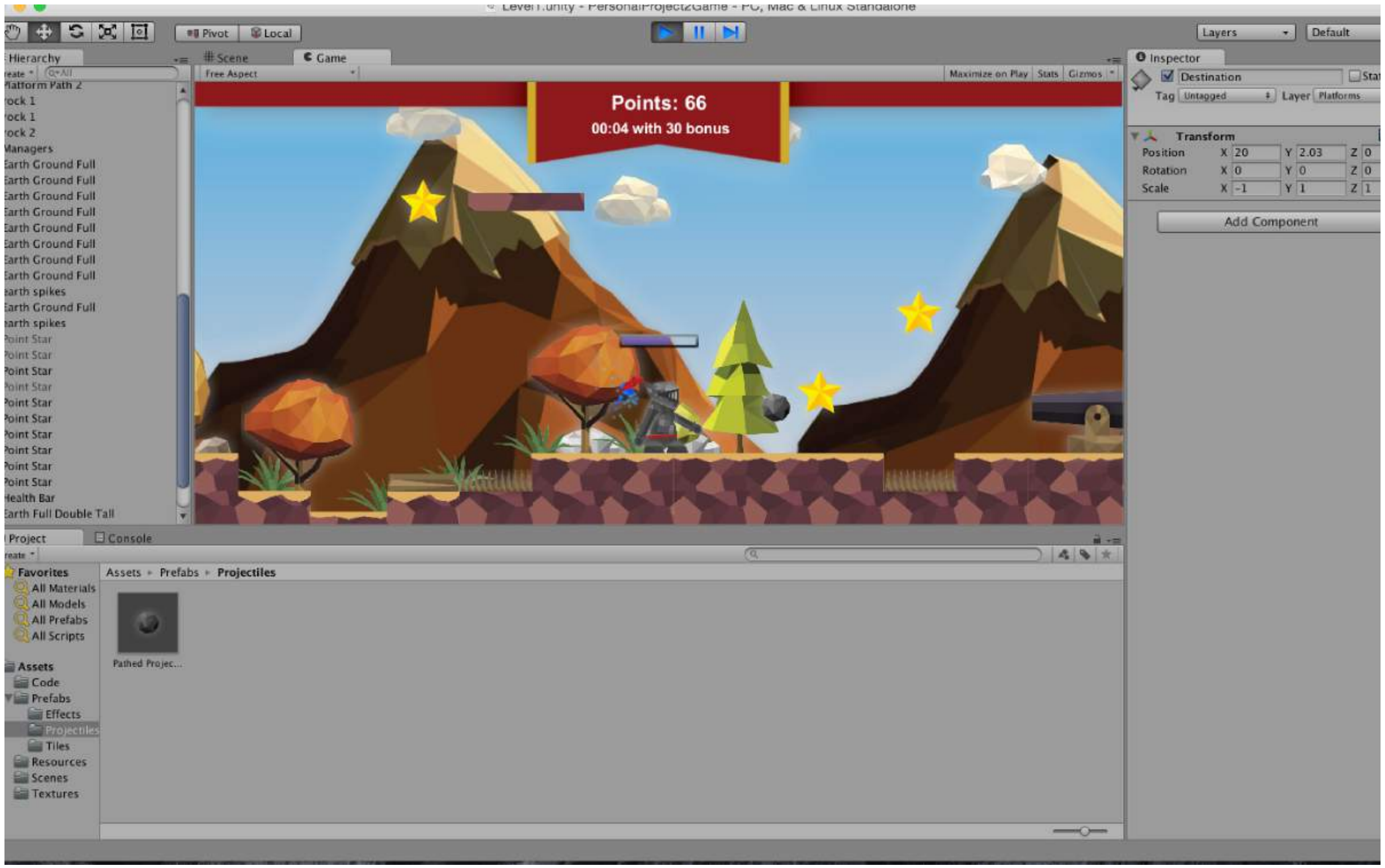

PathedProjectile.cs
PathedProjectileSpawner.cs
GiveDamageToPlayer.cs
GiveDamageToPlayer ▸ OnTriggerEnter2D (Collider2D other)

```
using UnityEngine;
using System.Collections;

public class GiveDamageToPlayer : MonoBehaviour
{
    public int DamageToGive = 10;

    private Vector2
        _lastPosition,
        _velocity;

    public void LateUpdate()
    {
        _velocity = (_lastPosition - (Vector2)transform.position) / Time.deltaTime;
        _lastPosition = transform.position;
    }

    public void OnTriggerEnter2D(Collider2D other)
    {
        var player = other.GetComponent<Player>();
        if (player == null)
            return;

        player.TakeDamage(DamageToGive);
        var controller = player.GetComponent<CharacterController2D>();
        var totalVelocity = controller.Velocity + _velocity;

        controller.SetForce(new Vector2(
            -1 * Mathf.Sign(totalVelocity.x) * Mathf.Clamp(Mathf.Abs(totalVelocity.x) * 6, 10, 40),
            -1 * Mathf.Sign(totalVelocity.y) * Mathf.Clamp(Mathf.Abs(totalVelocity.y) * 6, 5, 30)));
    }
}
```

GAME DEVELOPMENT

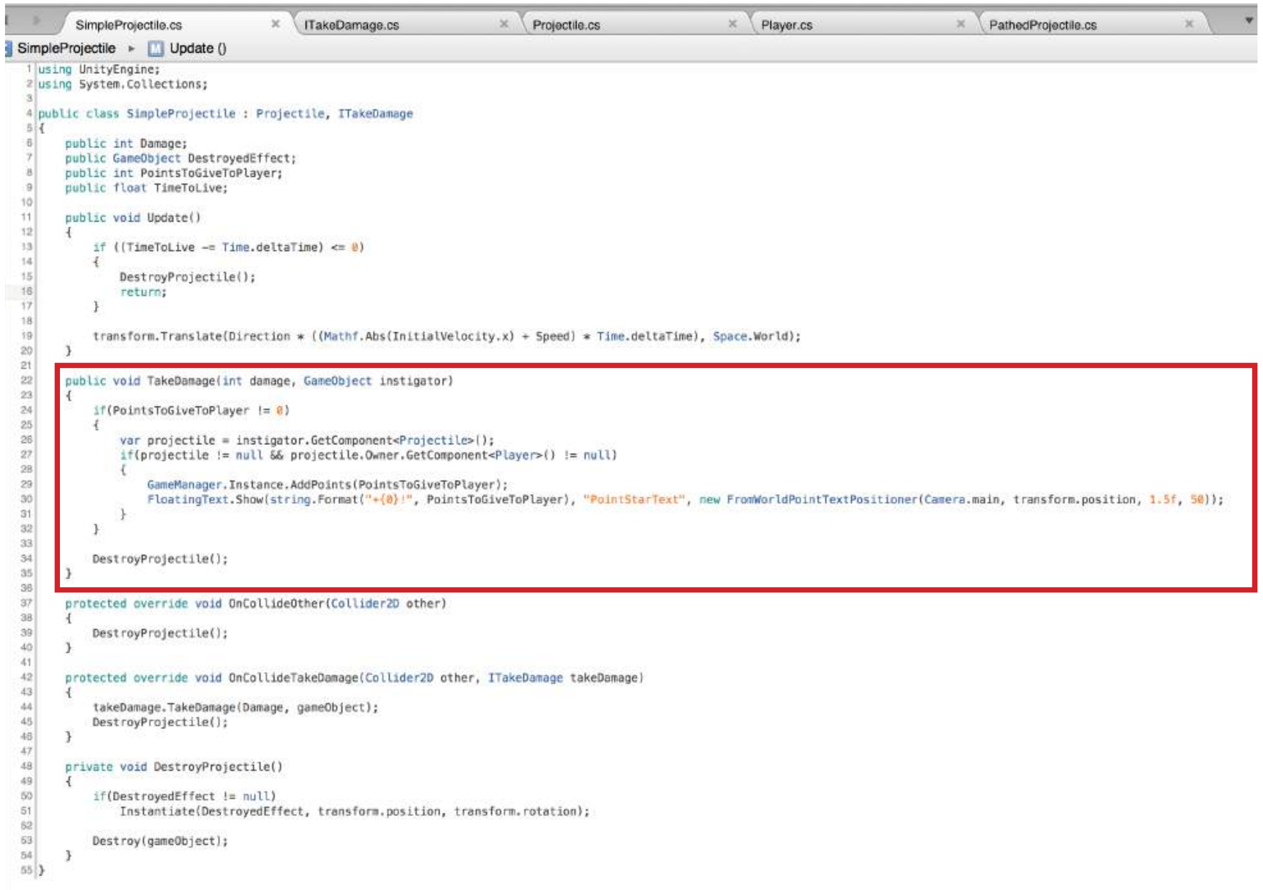
- PATHED PROJECTILES

When I go back into Unity and press play, I can see that the cannon does fire a cannon ball and this is the exact thing I want to happen. The player cannot destroy the ball yet but I want text to appear and points to be added to their score when doing so, I will do this next.



- PATHED PROJECTILES

The Simple Projectile code is updated, so, for when a projectile is fired and the object is destroyed, points will be added to the score and text will appear. It works a similar way to the floating star text, as the code is very similar to that.

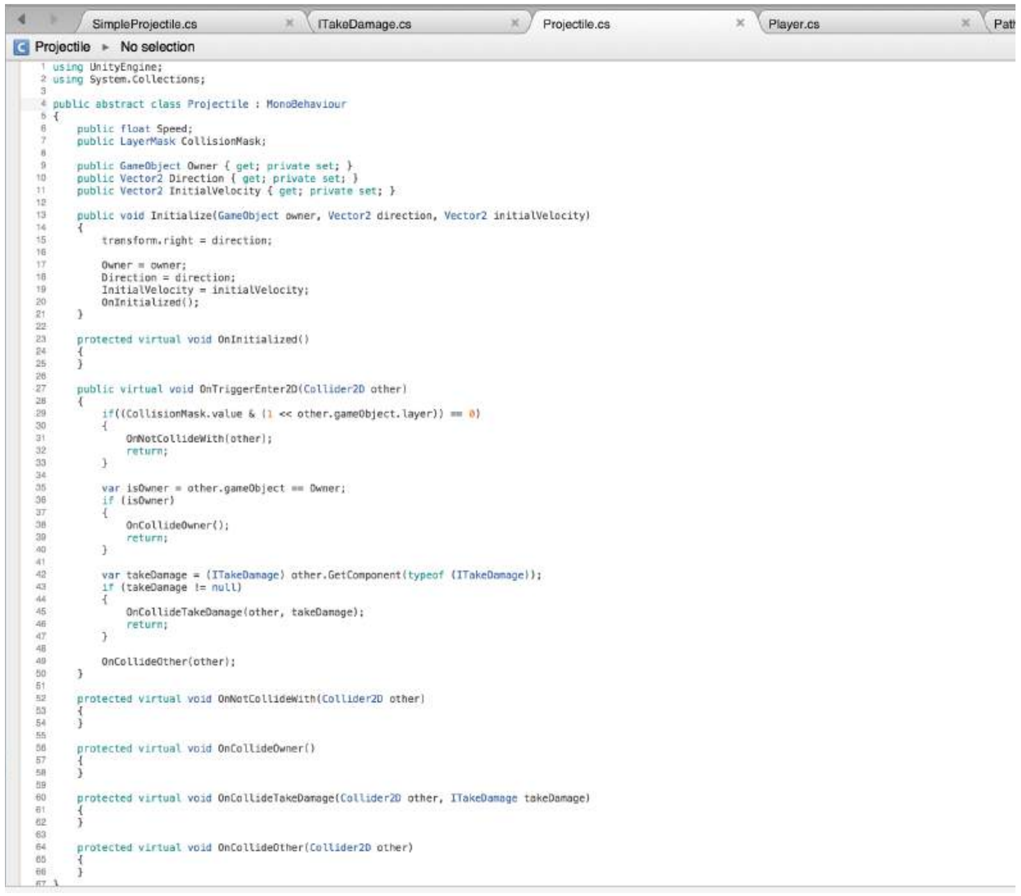


```
using UnityEngine;
using System.Collections;

public class SimpleProjectile : Projectile, ITakeDamage
{
    public int Damage;
    public GameObject DestroyedEffect;
    public int PointsToGiveToPlayer;
    public float TimeToLive;

    public void Update()
    {
        if ((TimeToLive -= Time.deltaTime) <= 0)
        {
            DestroyProjectile();
            return;
        }

        transform.Translate(Direction * ((Mathf.Abs(InitialVelocity.x) + Speed) * Time.deltaTime), Space.World);
    }

    public void TakeDamage(int damage, GameObject instigator)
    {
        if(PointsToGiveToPlayer != 0)
        {
            var projectile = instigator.GetComponent<Projectile>();
            if(projectile != null && projectile.Owner.GetComponent<Player>() != null)
            {
                GameManager.Instance.AddPoints(PointsToGiveToPlayer);
                FloatingText.Show(string.Format("+{0}!", PointsToGiveToPlayer), "PointStarText", new FromWorldPointTextPositioner(Camera.main, transform.position, 1.5f, 50));
            }
        }

        DestroyProjectile();
    }

    protected override void OnCollideOther(Collider2D other)
    {
        DestroyProjectile();
    }

    protected override void OnCollideTakeDamage(Collider2D other, ITakeDamage takeDamage)
    {
        takeDamage.TakeDamage(Damage, gameObject);
        DestroyProjectile();
    }

    private void DestroyProjectile()
    {
        if(DestroyedEffect != null)
            Instantiate(DestroyedEffect, transform.position, transform.rotation);

        Destroy(gameObject);
    }
}
```

GAME DEVELOPMENT

- PLAYER PROJECTILES

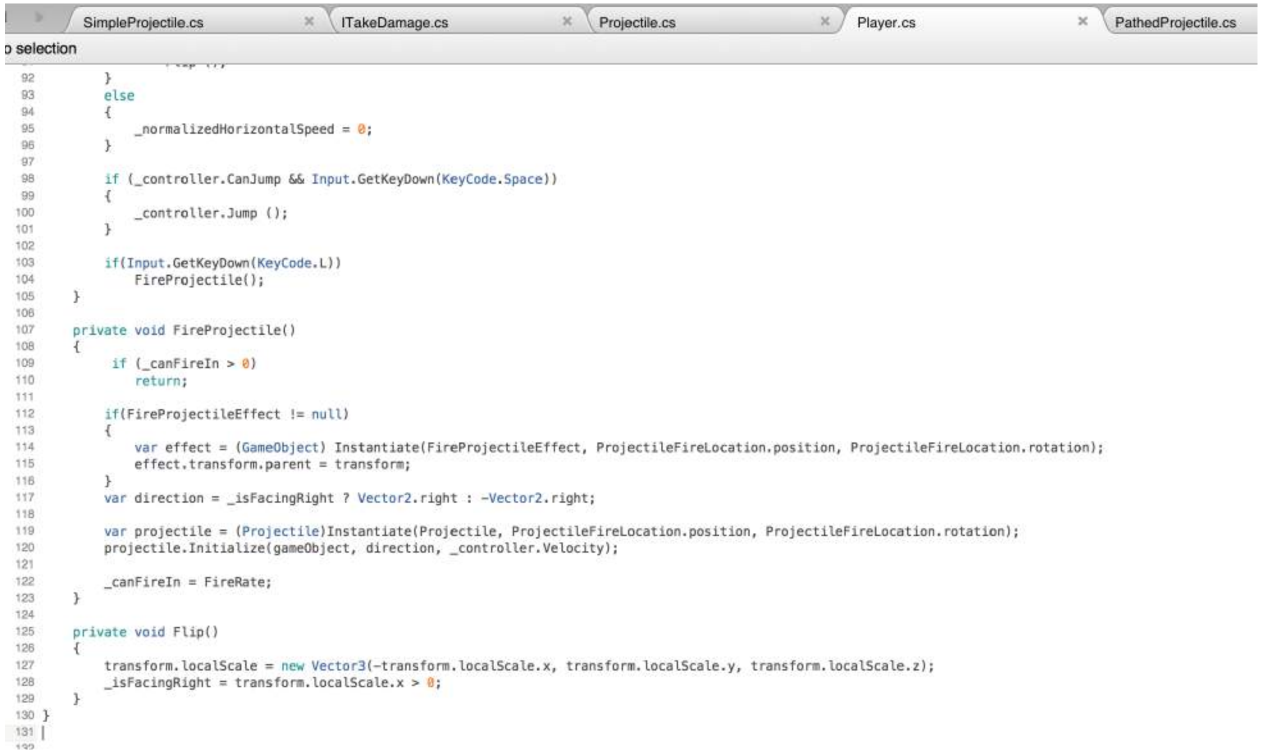
This is the code for the player projectile itself. If it is thrown by the player in any direction, the enemy projectile will take damage and be destroyed.



```csharp
using UnityEngine;
using System.Collections;

public abstract class Projectile : MonoBehaviour
{
    public float Speed;
    public LayerMask CollisionMask;

    public GameObject Owner { get; private set; }
    public Vector2 Direction { get; private set; }
    public Vector2 InitialVelocity { get; private set; }

    public void Initialize(GameObject owner, Vector2 direction, Vector2 initialVelocity)
    {
        transform.right = direction;

        Owner = owner;
        Direction = direction;
        InitialVelocity = initialVelocity;
        OnInitialized();
    }

    protected virtual void OnInitialized()
    {
    }

    public virtual void OnTriggerEnter2D(Collider2D other)
    {
        if((CollisionMask.value & (1 << other.gameObject.layer)) == 0)
        {
            OnNotCollideWith(other);
            return;
        }

        var isOwner = other.gameObject == Owner;
        if (isOwner)
        {
            OnCollideOwner();
            return;
        }

        var takeDamage = (ITakeDamage) other.GetComponent(typeof (ITakeDamage));
        if (takeDamage != null)
        {
            OnCollideTakeDamage(other, takeDamage);
            return;
        }

        OnCollideOther(other);
    }

    protected virtual void OnNotCollideWith(Collider2D other)
    {
    }

    protected virtual void OnCollideOwner()
    {
    }

    protected virtual void OnCollideTakeDamage(Collider2D other, ITakeDamage takeDamage)
    {
    }

    protected virtual void OnCollideOther(Collider2D other)
    {
    }
}
```

- PLAYER PROJECTILES

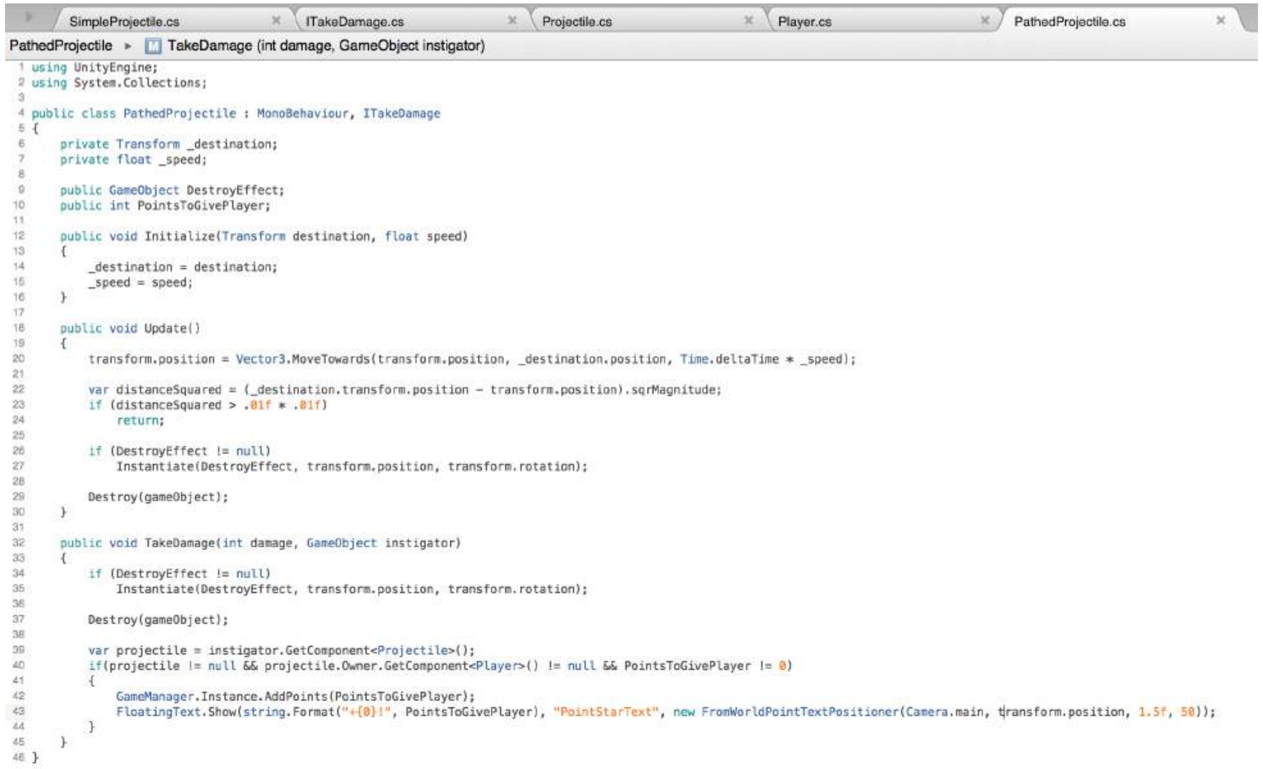
Back in the Player script, the code has been updated so that when a projectile has been fired out of its hand it is thrown from the keyboard input, I have inputted "L" for this command but will change it when I come to the actual glove controller.



```
        }
        else
        {
            _normalizedHorizontalSpeed = 0;
        }

        if (_controller.CanJump && Input.GetKeyDown(KeyCode.Space))
        {
            _controller.Jump ();
        }

        if(Input.GetKeyDown(KeyCode.L))
            FireProjectile();
    }

    private void FireProjectile()
    {
         if (_canFireIn > 0)
            return;

        if(FireProjectileEffect != null)
        {
            var effect = (GameObject) Instantiate(FireProjectileEffect, ProjectileFireLocation.position, ProjectileFireLocation.rotation);
            effect.transform.parent = transform;
        }
        var direction = _isFacingRight ? Vector2.right : -Vector2.right;

        var projectile = (Projectile)Instantiate(Projectile, ProjectileFireLocation.position, ProjectileFireLocation.rotation);
        projectile.Initialize(gameObject, direction, _controller.Velocity);

        _canFireIn = FireRate;
    }

    private void Flip()
    {
        transform.localScale = new Vector3(-transform.localScale.x, transform.localScale.y, transform.localScale.z);
        _isFacingRight = transform.localScale.x > 0;
    }
}
```

- PLAYER PROJECTILES

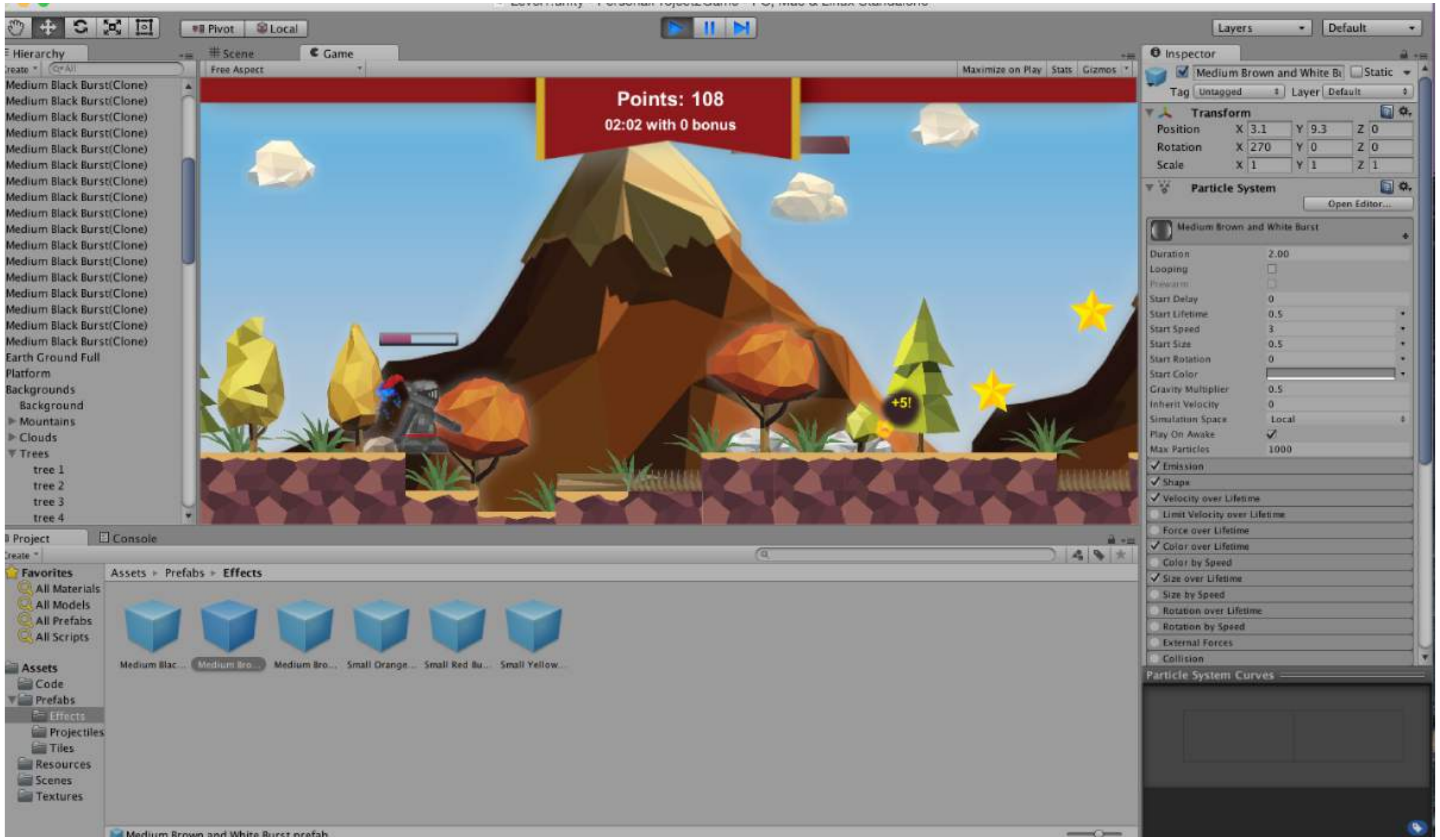
The Pathed Projectile code has now got a few lines of code saying to Unity when the object has been destroyed by the player, the player receives points that is then added to their current score.



```
using UnityEngine;
using System.Collections;

public class PathedProjectile : MonoBehaviour, ITakeDamage
{
    private Transform _destination;
    private float _speed;

    public GameObject DestroyEffect;
    public int PointsToGivePlayer;

    public void Initialize(Transform destination, float speed)
    {
        _destination = destination;
        _speed = speed;
    }

    public void Update()
    {
        transform.position = Vector3.MoveTowards(transform.position, _destination.position, Time.deltaTime * _speed);

        var distanceSquared = (_destination.transform.position - transform.position).sqrMagnitude;
        if (distanceSquared > .01f * .01f)
            return;

        if (DestroyEffect != null)
            Instantiate(DestroyEffect, transform.position, transform.rotation);

        Destroy(gameObject);
    }

    public void TakeDamage(int damage, GameObject instigator)
    {
        if (DestroyEffect != null)
            Instantiate(DestroyEffect, transform.position, transform.rotation);

        Destroy(gameObject);

        var projectile = instigator.GetComponent<Projectile>();
        if(projectile != null && projectile.Owner.GetComponent<Player>() != null && PointsToGivePlayer != 0)
        {
            GameManager.Instance.AddPoints(PointsToGivePlayer);
            FloatingText.Show(string.Format("+{0}!", PointsToGivePlayer), "PointStarText", new FromWorldPointTextPositioner(Camera.main, transform.position, 1.5f, 50));
        }
    }
}
```

- PLAYER PROJECTILES

Back in Unity I can test this code out. When I play the game and throw an axe, then it collides with the cannon ball, it is destroyed and the player is awarded points, which means this is exactly what I want so, the players axe projectile is working as it should.



Inspector

Medium Brown and White Bu | Static

Tag: Untagged | Layer: Default

Transform

	X	Y	Z
Position	3.1	9.3	0
Rotation	270	0	0
Scale	1	1	1

Particle System — Open Editor...

Medium Brown and White Burst

Duration	2.00
Looping	☐
Prewarm	☐
Start Delay	0
Start Lifetime	0.5
Start Speed	3
Start Size	0.5
Start Rotation	0
Start Color	
Gravity Multiplier	0.5
Inherit Velocity	0
Simulation Space	Local
Play On Awake	✓
Max Particles	1000

✓ Emission
✓ Shape
✓ Velocity over Lifetime
Limit Velocity over Lifetime
Force over Lifetime
✓ Color over Lifetime
Color by Speed
✓ Size over Lifetime
Size by Speed
Rotation over Lifetime
Rotation by Speed
External Forces
Collision

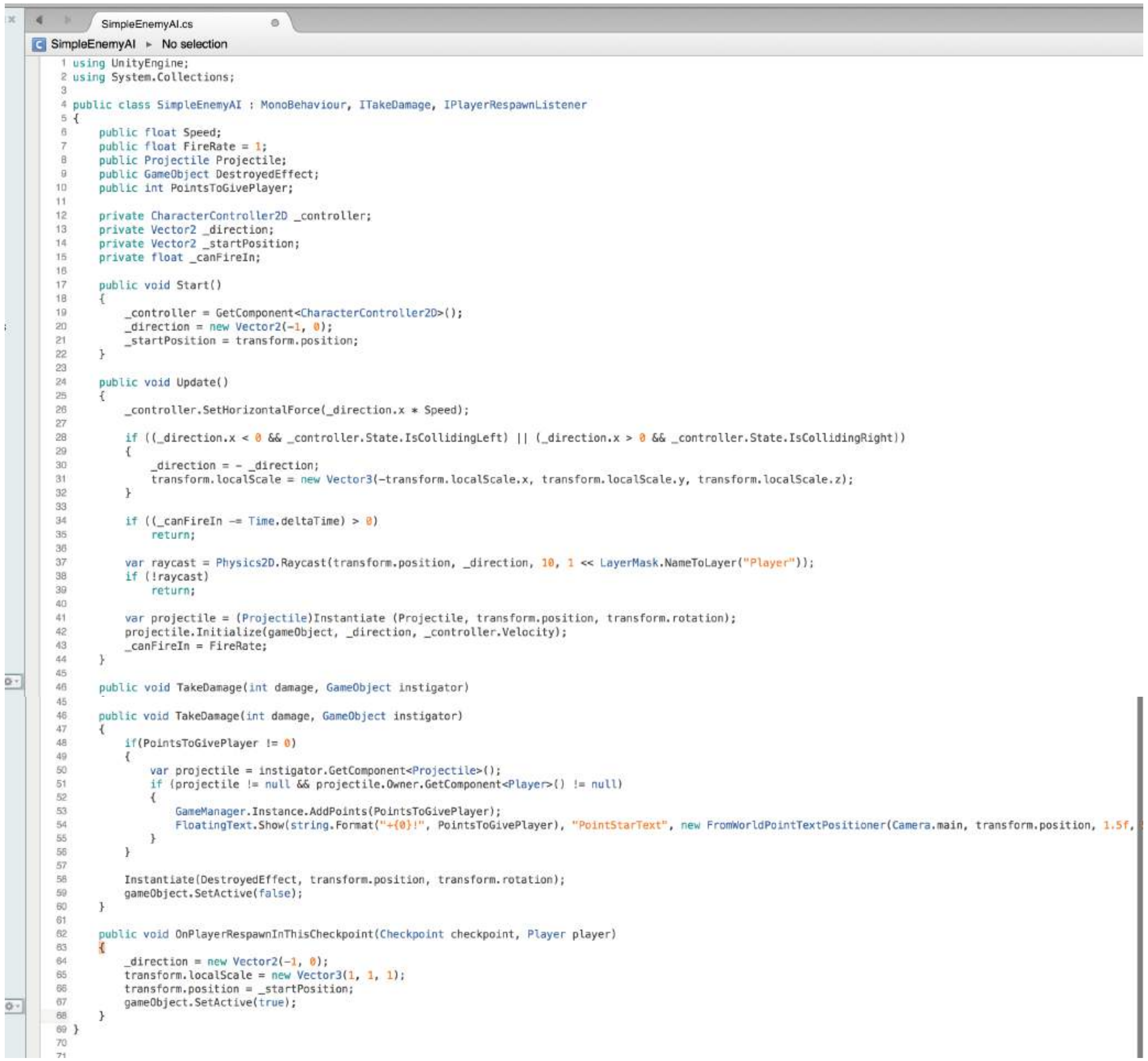
Particle System Curves

- ENEMY AI

Part of my game will feature a dragon enemy that will eject fire from its mouth and damage the player if they are hit.

I created an enemy AI script that will allow the dragon to move backwards and forwards between a range and fire a projectile when the player is near them, and when it hits the player, they will take damage.

However, when a projectile is fired from the player, if it hits the dragon, the dragon will die.



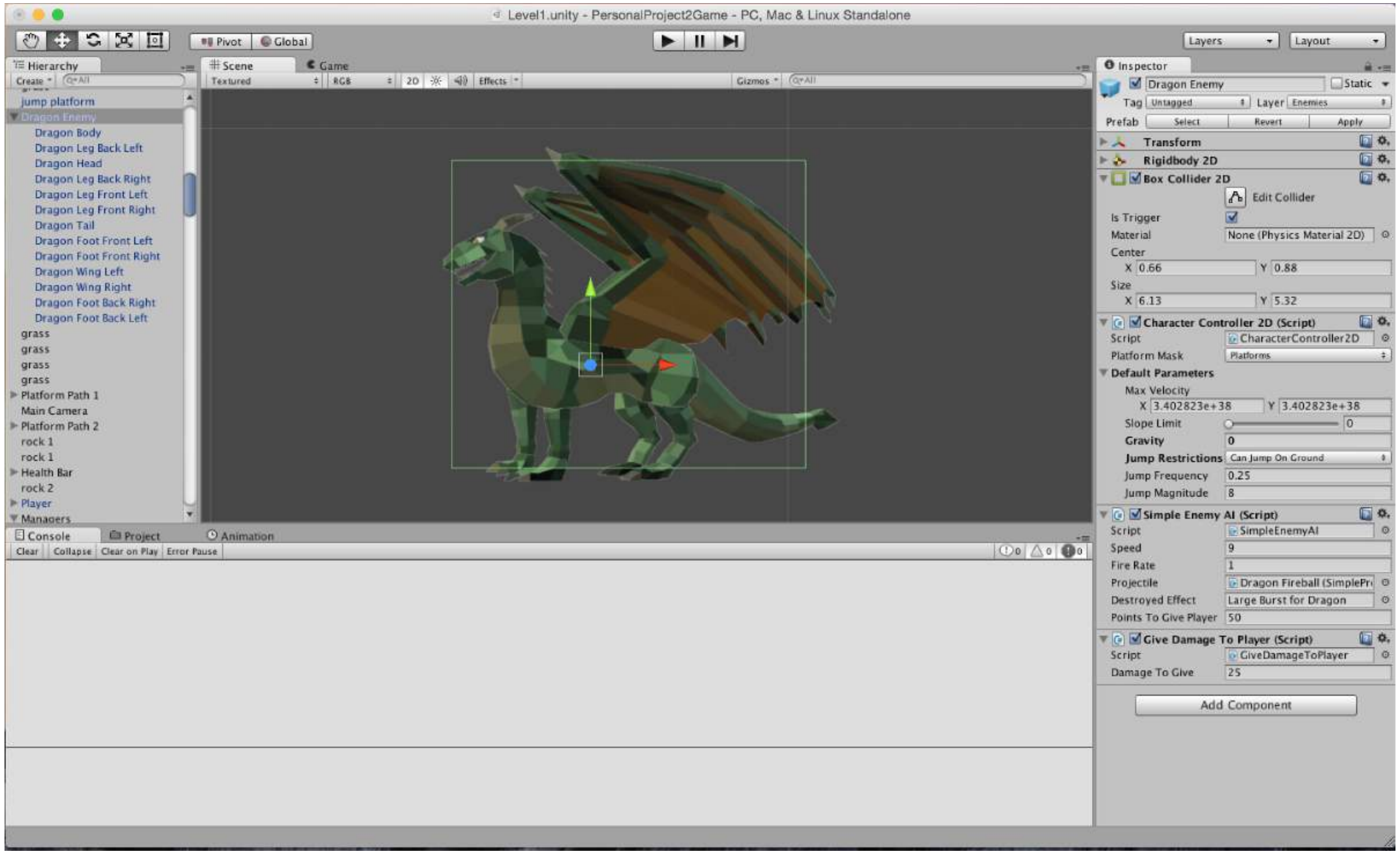
```
using UnityEngine;
using System.Collections;

public class SimpleEnemyAI : MonoBehaviour, ITakeDamage, IPlayerRespawnListener
{
    public float Speed;
    public float FireRate = 1;
    public Projectile Projectile;
    public GameObject DestroyedEffect;
    public int PointsToGivePlayer;

    private CharacterController2D _controller;
    private Vector2 _direction;
    private Vector2 _startPosition;
    private float _canFireIn;

    public void Start()
    {
        _controller = GetComponent<CharacterController2D>();
        _direction = new Vector2(-1, 0);
        _startPosition = transform.position;
    }

    public void Update()
    {
        _controller.SetHorizontalForce(_direction.x * Speed);

        if ((_direction.x < 0 && _controller.State.IsCollidingLeft) || (_direction.x > 0 && _controller.State.IsCollidingRight))
        {
            _direction = - _direction;
            transform.localScale = new Vector3(-transform.localScale.x, transform.localScale.y, transform.localScale.z);
        }

        if ((_canFireIn -= Time.deltaTime) > 0)
            return;

        var raycast = Physics2D.Raycast(transform.position, _direction, 10, 1 << LayerMask.NameToLayer("Player"));
        if (!raycast)
            return;

        var projectile = (Projectile)Instantiate (Projectile, transform.position, transform.rotation);
        projectile.Initialize(gameObject, _direction, _controller.Velocity);
        _canFireIn = FireRate;
    }

    public void TakeDamage(int damage, GameObject instigator)

    public void TakeDamage(int damage, GameObject instigator)
    {
        if(PointsToGivePlayer != 0)
        {
            var projectile = instigator.GetComponent<Projectile>();
            if (projectile != null && projectile.Owner.GetComponent<Player>() != null)
            {
                GameManager.Instance.AddPoints(PointsToGivePlayer);
                FloatingText.Show(string.Format("+{0}!", PointsToGivePlayer), "PointStarText", new FromWorldPointTextPositioner(Camera.main, transform.position, 1.5f,
            }
        }

        Instantiate(DestroyedEffect, transform.position, transform.rotation);
        gameObject.SetActive(false);
    }

    public void OnPlayerRespawnInThisCheckpoint(Checkpoint checkpoint, Player player)
    {
        _direction = new Vector2(-1, 0);
        transform.localScale = new Vector3(1, 1, 1);
        transform.position = _startPosition;
        gameObject.SetActive(true);
    }
}
```

- ENEMY AI

Back in Unity I imported my sprite dragon enemy, assemble him together and assign components.

This sprite will be built similar to the player. A "RigidBody2D" is added, make kinematic as well as the same "CharacterController2D" script.

The only difference I make is by adding the new "SimpleEnemyAI" script as well as the "GiveDamageToPlayer" script previously used in the cannon ball sprite.

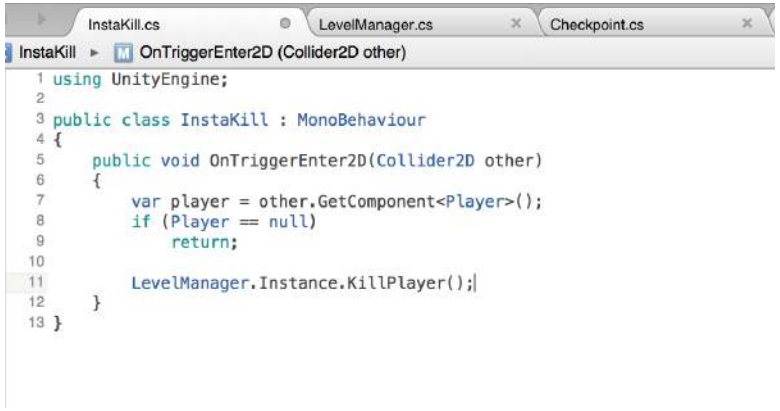


```
Dragon Enemy                         Static
Tag Untagged        Layer Enemies
Prefab   Select   Revert   Apply
Transform
Rigidbody 2D
Box Collider 2D
                  Edit Collider
Is Trigger        ✓
Material          None (Physics Material 2D)
Center
  X 0.66            Y 0.88
Size
  X 6.13            Y 5.32
Character Controller 2D (Script)
Script            CharacterController2D
Platform Mask     Platforms
Default Parameters
  Max Velocity
    X 3.402823e+38  Y 3.402823e+38
  Slope Limit       0
  Gravity           0
  Jump Restrictions Can Jump On Ground
  Jump Frequency    0.25
  Jump Magnitude    8
Simple Enemy AI (Script)
Script            SimpleEnemyAI
Speed             9
Fire Rate         1
Projectile        Dragon Fireball (SimplePr
Destroyed Effect  Large Burst for Dragon
Points To Give Player 50
Give Damage To Player (Script)
Script            GiveDamageToPlayer
Damage To Give    25
Add Component
```

GAME DEVELOPMENT

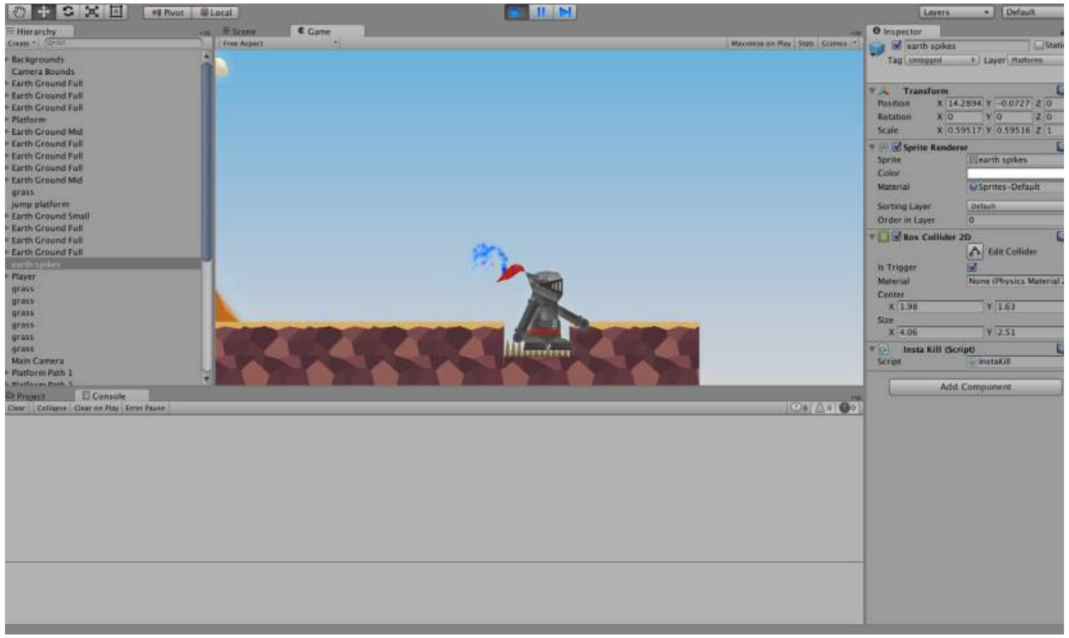
- INSTAKILL TRAPS

I said from the begging that I wanted traps etc. in my game. The "instakill" code for traps will act as a component to kill the player if they collide with them.



```
using UnityEngine;

public class InstaKill : MonoBehaviour
{
    public void OnTriggerEnter2D(Collider2D other)
    {
        var player = other.GetComponent<Player>();
        if (Player == null)
            return;

        LevelManager.Instance.KillPlayer();
    }
}
```

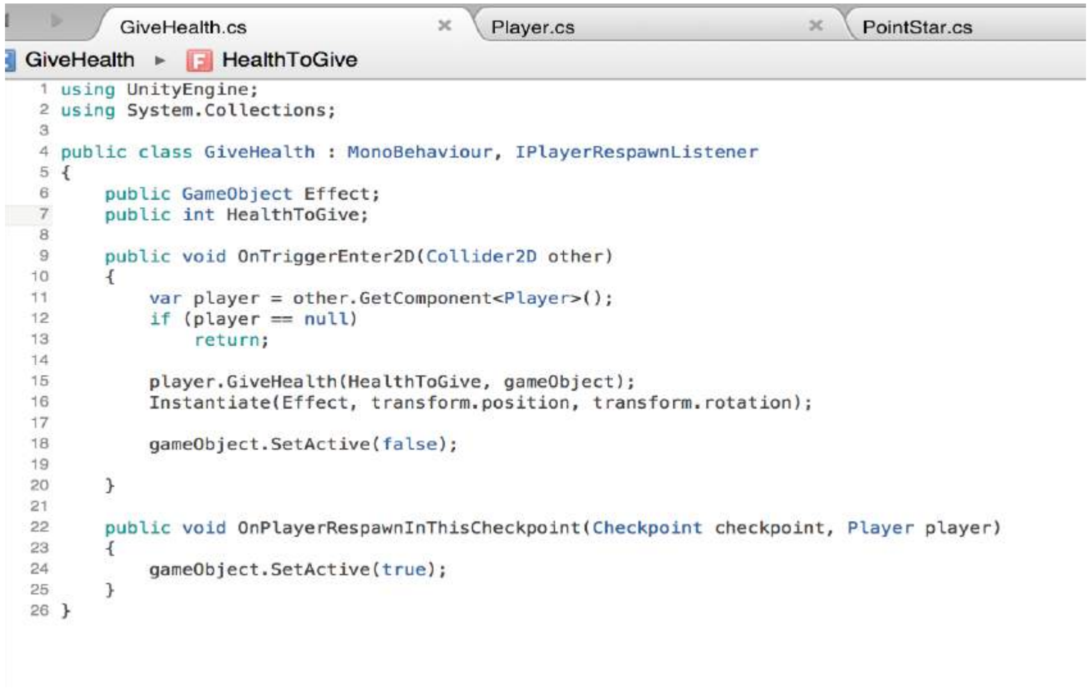
When I test it, the instakill with checkpoints are working, this is correct! the player falls through the map and will respawn at the last checkpoint.



```
earth spikes
Tag Untagged   Layer Platforms
Transform
Position   X 14.2894  Y -0.0727  Z 0
Rotation   X 0        Y 0        Z 0
Scale      X 0.59517  Y 0.59516  Z 1
Sprite Renderer
Sprite          earth spikes
Color
Material        Sprites-Default
Sorting Layer   Default
Order in Layer  0
Box Collider 2D
Edit Collider
Is Trigger      ✓
Material        None (Physics Material 2D)
Center
  X 1.98   Y 1.63
Size
  X 4.06   Y 2.51
Insta Kill (Script)
Script          InstaKill
```

- HEALTH PACKS

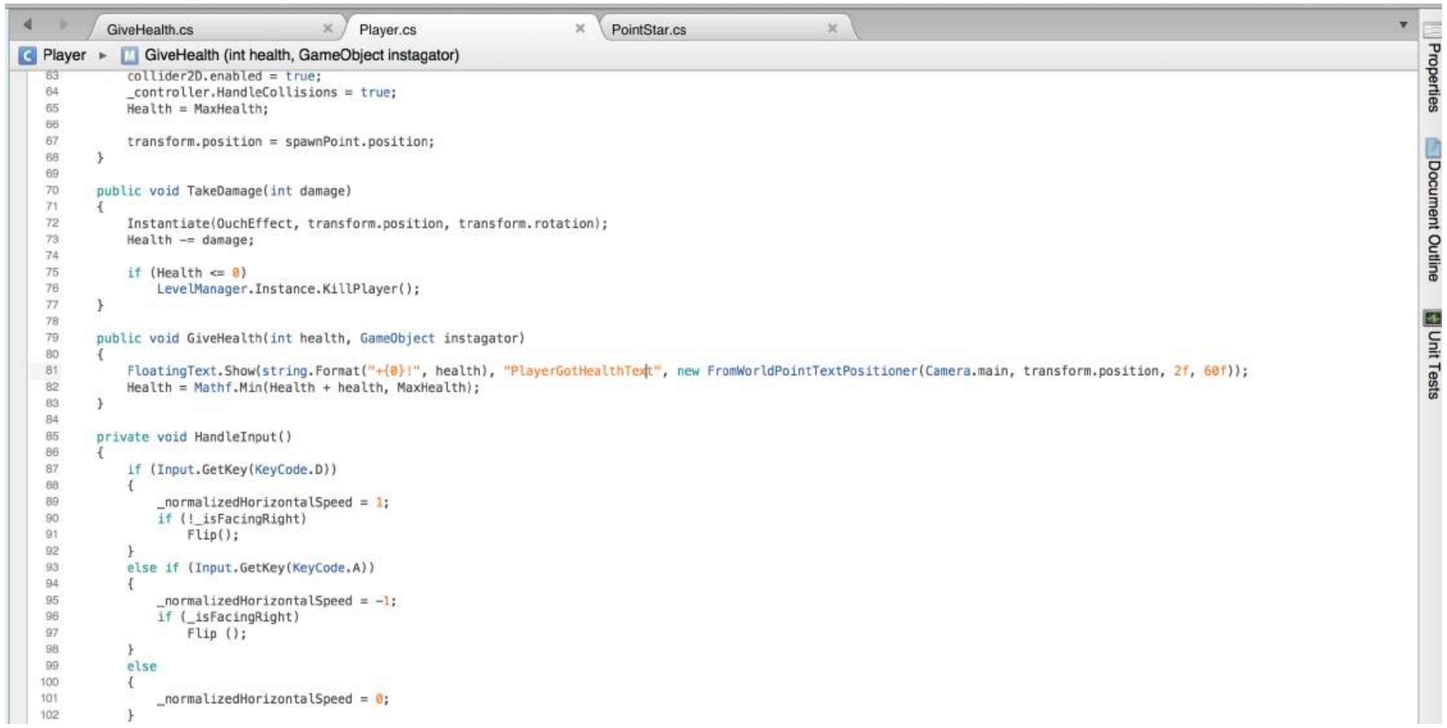
This script is written for the health pack, it works similar to point star script, when the player collides with the heart collectable, the player will gain health back which is a great game mechanic so the player can carry on though the level with a higher chance of succeeding if the take damage from projectiles.



```
using UnityEngine;
using System.Collections;

public class GiveHealth : MonoBehaviour, IPlayerRespawnListener
{
    public GameObject Effect;
    public int HealthToGive;

    public void OnTriggerEnter2D(Collider2D other)
    {
        var player = other.GetComponent<Player>();
        if (player == null)
            return;

        player.GiveHealth(HealthToGive, gameObject);
        Instantiate(Effect, transform.position, transform.rotation);

        gameObject.SetActive(false);

    }

    public void OnPlayerRespawnInThisCheckpoint(Checkpoint checkpoint, Player player)
    {
        gameObject.SetActive(true);
    }
}
```

- HEALTH PACKS

Back in the Player script, the new lines of code I have written shows how the player can gain health back once a heart is obtained, but no more than max health (100) because the max health the player can have is 100.

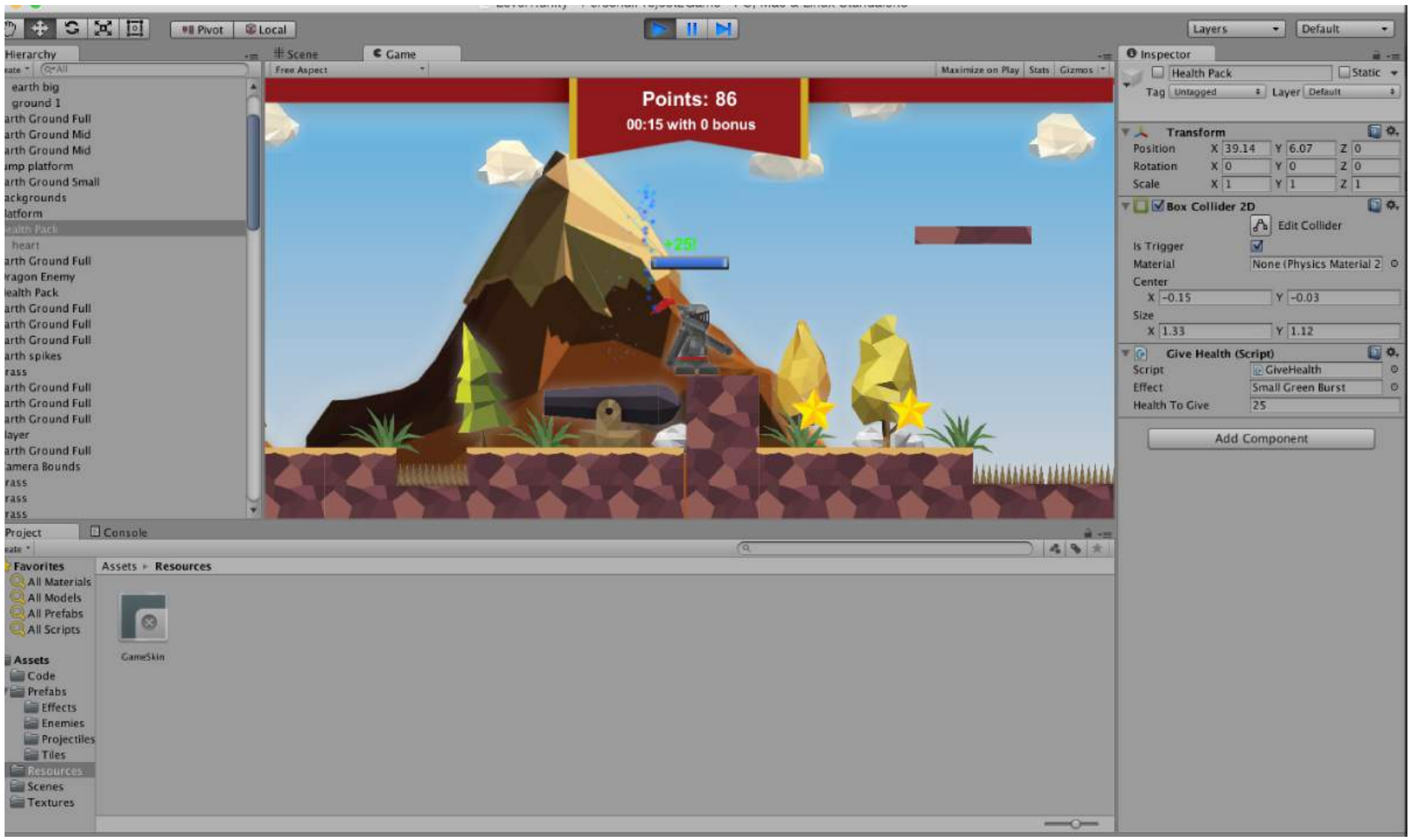


```
        collider2D.enabled = true;
        _controller.HandleCollisions = true;
        Health = MaxHealth;

        transform.position = spawnPoint.position;
    }

    public void TakeDamage(int damage)
    {
        Instantiate(OuchEffect, transform.position, transform.rotation);
        Health -= damage;

        if (Health <= 0)
            LevelManager.Instance.KillPlayer();
    }

    public void GiveHealth(int health, GameObject instagator)
    {
        FloatingText.Show(string.Format("+{0}!", health), "PlayerGotHealthText", new FromWorldPointTextPositioner(Camera.main, transform.position, 2f, 60f));
        Health = Mathf.Min(Health + health, MaxHealth);
    }

    private void HandleInput()
    {
        if (Input.GetKey(KeyCode.D))
        {
            _normalizedHorizontalSpeed = 1;
            if (!_isFacingRight)
                Flip();
        }
        else if (Input.GetKey(KeyCode.A))
        {
            _normalizedHorizontalSpeed = -1;
            if (_isFacingRight)
                Flip ();
        }
        else
        {
            _normalizedHorizontalSpeed = 0;
        }
```

GAME DEVELOPMENT

- HEALTH PACKS

When I go back into Unity and press play, I collect a heart and I get 25 health back on the player, I know this because of the health bar and the floating text that says "+25!" and the player gets no more than 100 which is exactly what I want for my game.


Health Pack ☐ Static
Tag Untagged Layer Default

Transform
Position X 39.14 Y 6.07 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

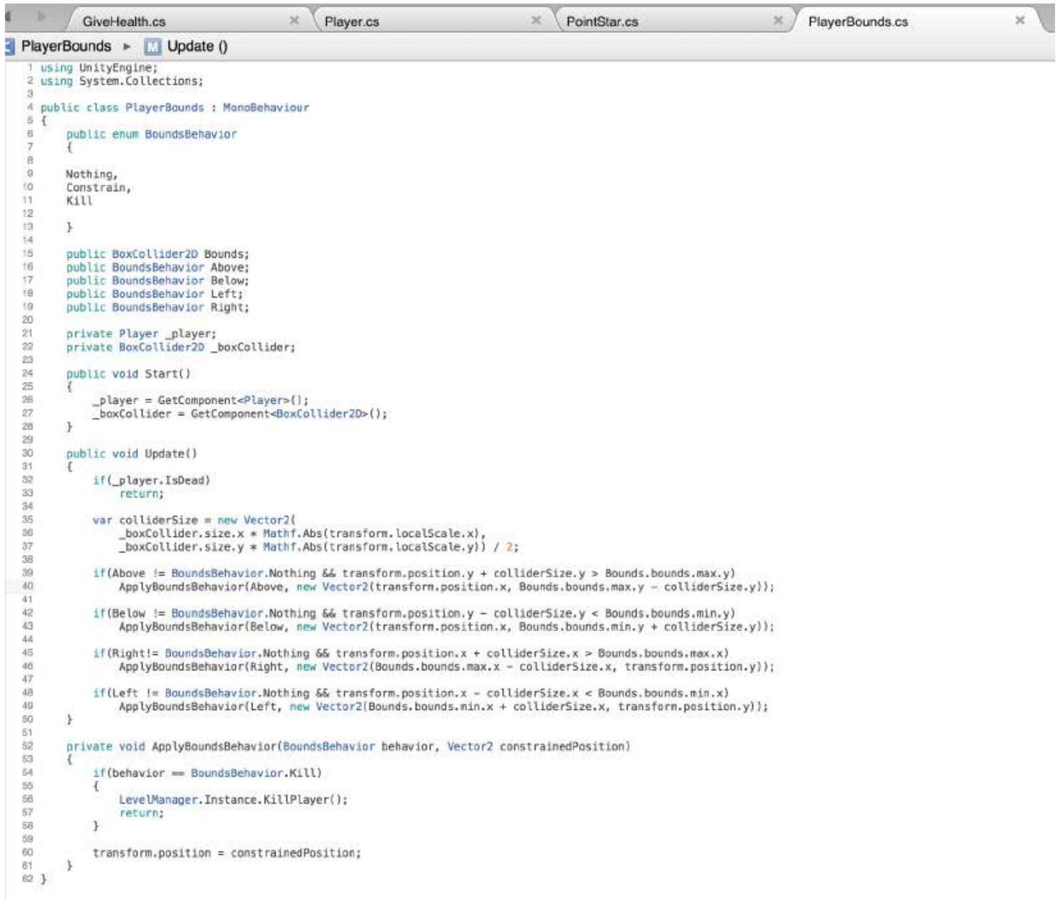
☑ Box Collider 2D
Edit Collider
Is Trigger ☑
Material None (Physics Material 2
Center X -0.15 Y -0.03
Size X 1.33 Y 1.12

Give Health (Script)
Script GiveHealth
Effect Small Green Burst
Health To Give 25

- MOVEMENT CONSTRAINTS

While playing my game I noticed that I could fall off my level if I chose to go left instead of right, through the level.

This code is used for setting camera constraints so the player cannot fall off the edge of the scene, he cant fall off left or right, he will die if he touches the bottom of the screen too, which is a great game mechanic so the player is bounded to the level world and cant fall off or glitch out while playing.



```
using UnityEngine;
using System.Collections;

public class PlayerBounds : MonoBehaviour
{
    public enum BoundsBehavior
    {

    Nothing,
    Constrain,
    Kill

    }

    public BoxCollider2D Bounds;
    public BoundsBehavior Above;
    public BoundsBehavior Below;
    public BoundsBehavior Left;
    public BoundsBehavior Right;

    private Player _player;
    private BoxCollider2D _boxCollider;

    public void Start()
    {
        _player = GetComponent<Player>();
        _boxCollider = GetComponent<BoxCollider2D>();
    }

    public void Update()
    {
        if(_player.IsDead)
            return;

        var colliderSize = new Vector2(
            _boxCollider.size.x * Mathf.Abs(transform.localScale.x),
            _boxCollider.size.y * Mathf.Abs(transform.localScale.y)) / 2;

        if(Above != BoundsBehavior.Nothing && transform.position.y + colliderSize.y > Bounds.bounds.max.y)
            ApplyBoundsBehavior(Above, new Vector2(transform.position.x, Bounds.bounds.max.y - colliderSize.y));

        if(Below != BoundsBehavior.Nothing && transform.position.y - colliderSize.y < Bounds.bounds.min.y)
            ApplyBoundsBehavior(Below, new Vector2(transform.position.x, Bounds.bounds.min.y + colliderSize.y));

        if(Right!= BoundsBehavior.Nothing && transform.position.x + colliderSize.x > Bounds.bounds.max.x)
            ApplyBoundsBehavior(Right, new Vector2(Bounds.bounds.max.x - colliderSize.x, transform.position.y));

        if(Left != BoundsBehavior.Nothing && transform.position.x - colliderSize.x < Bounds.bounds.min.x)
            ApplyBoundsBehavior(Left, new Vector2(Bounds.bounds.min.x + colliderSize.x, transform.position.y));
    }

    private void ApplyBoundsBehavior(BoundsBehavior behavior, Vector2 constrainedPosition)
    {
        if(behavior == BoundsBehavior.Kill)
        {
            LevelManager.Instance.KillPlayer();
            return;
        }

        transform.position = constrainedPosition;
    }
}
```

- AUDIO

I had found many .WAV files on the Unity asset store, which I could use for sound effects in my game, as they are open source and free!

I want sounds for:

- The background soundtrack
- Axe throw
- Jump platform
- Star pickup
- Health pickup
- Cannon fire

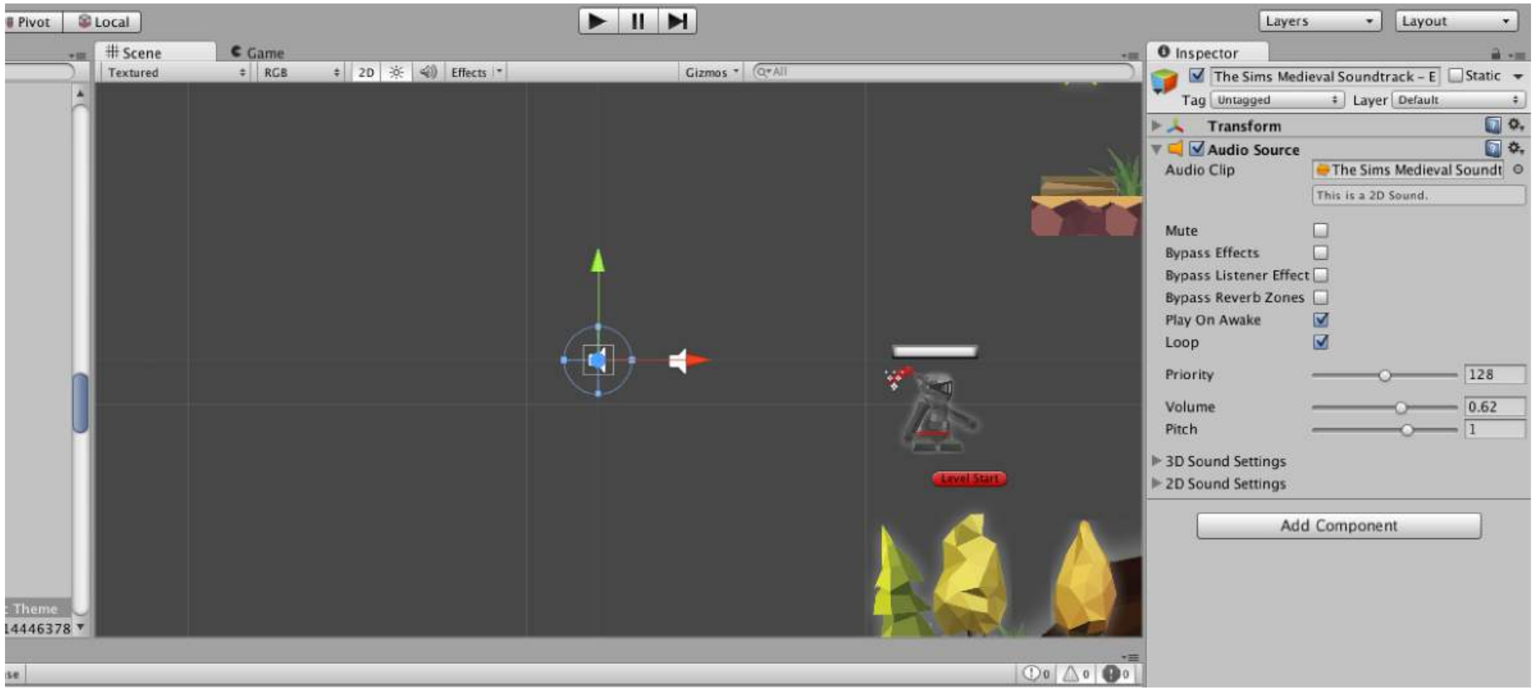
The background track was quite difficult to find because I wanted one to really fit the style of my game, I searched high and low for a medieval sounding track then I came across a track that was included in "The Sim's Medieval" game, The track can be found here on YouTube:

https://www.youtube.com/watch?v=Is5F4vQRZ3M

From feedback I chose to continue using this track as it fit my game well and it would start to bring my game to life a little as sound does a lot for a video game.

- AUDIO

Placing a sound in Unity is pretty simple really. All I need to do I import the sound file like any other file, then come to the scene view and drag it on, and it will play on loop as I progress through the level.



The Sims Medieval Soundtrack – E ☐ Static
Tag Untagged Layer Default

Audio Source
Audio Clip: The Sims Medieval Soundt
This is a 2D Sound.

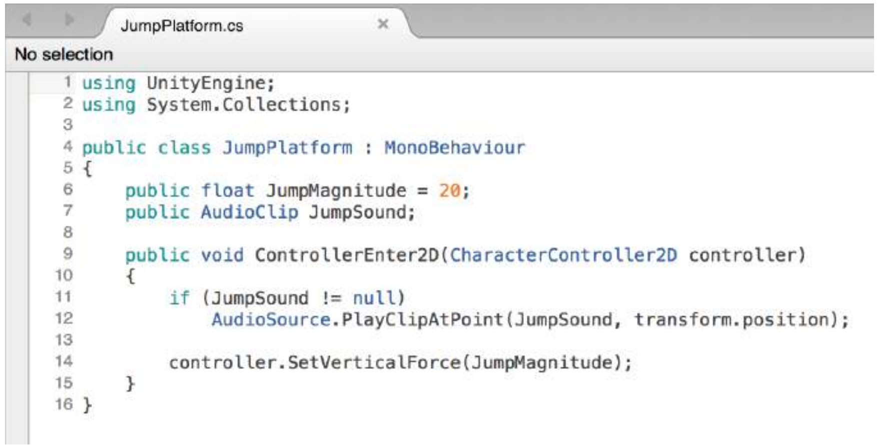
Mute	☐
Bypass Effects	☐
Bypass Listener Effect	☐
Bypass Reverb Zones	☐
Play On Awake	✓
Loop	✓
Priority	128
Volume	0.62
Pitch	1

3D Sound Settings
2D Sound Settings

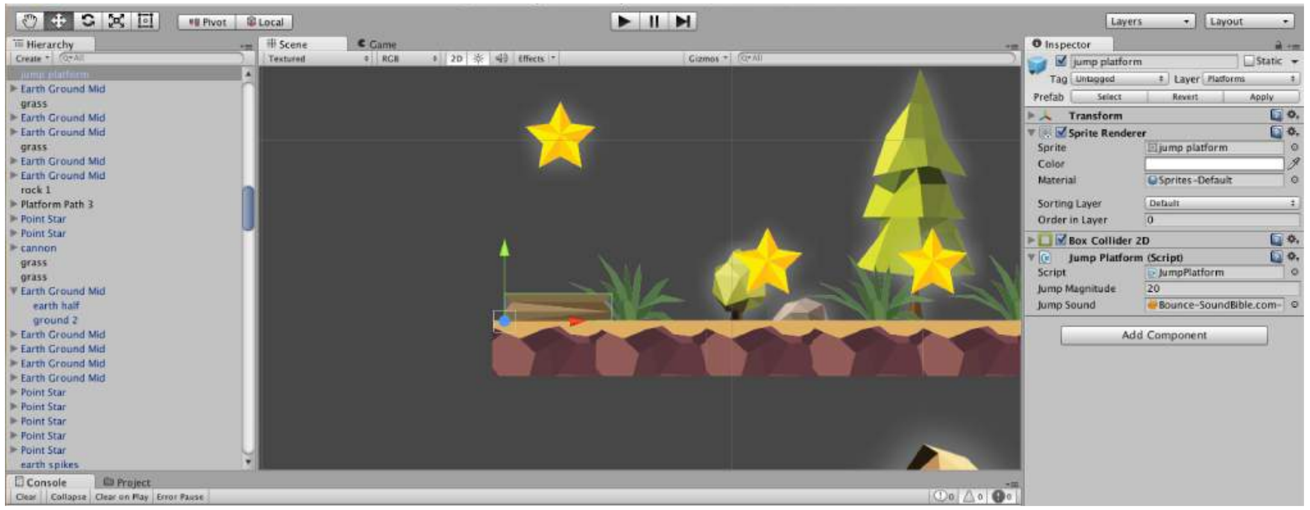
- AUDIO

To add audio to a specific object, it was a little different; in each code I used for a certain object, in this case the jump platform,

I had to define the audio source, once this line of code was implemented I was able to drag the audio clip that I would be using for that specific object into its properties, then once the player jumps on it the audio plays.



```
using UnityEngine;
using System.Collections;

public class JumpPlatform : MonoBehaviour
{
    public float JumpMagnitude = 20;
    public AudioClip JumpSound;

    public void ControllerEnter2D(CharacterController2D controller)
    {
        if (JumpSound != null)
            AudioSource.PlayClipAtPoint(JumpSound, transform.position);

        controller.SetVerticalForce(JumpMagnitude);
    }
}
```

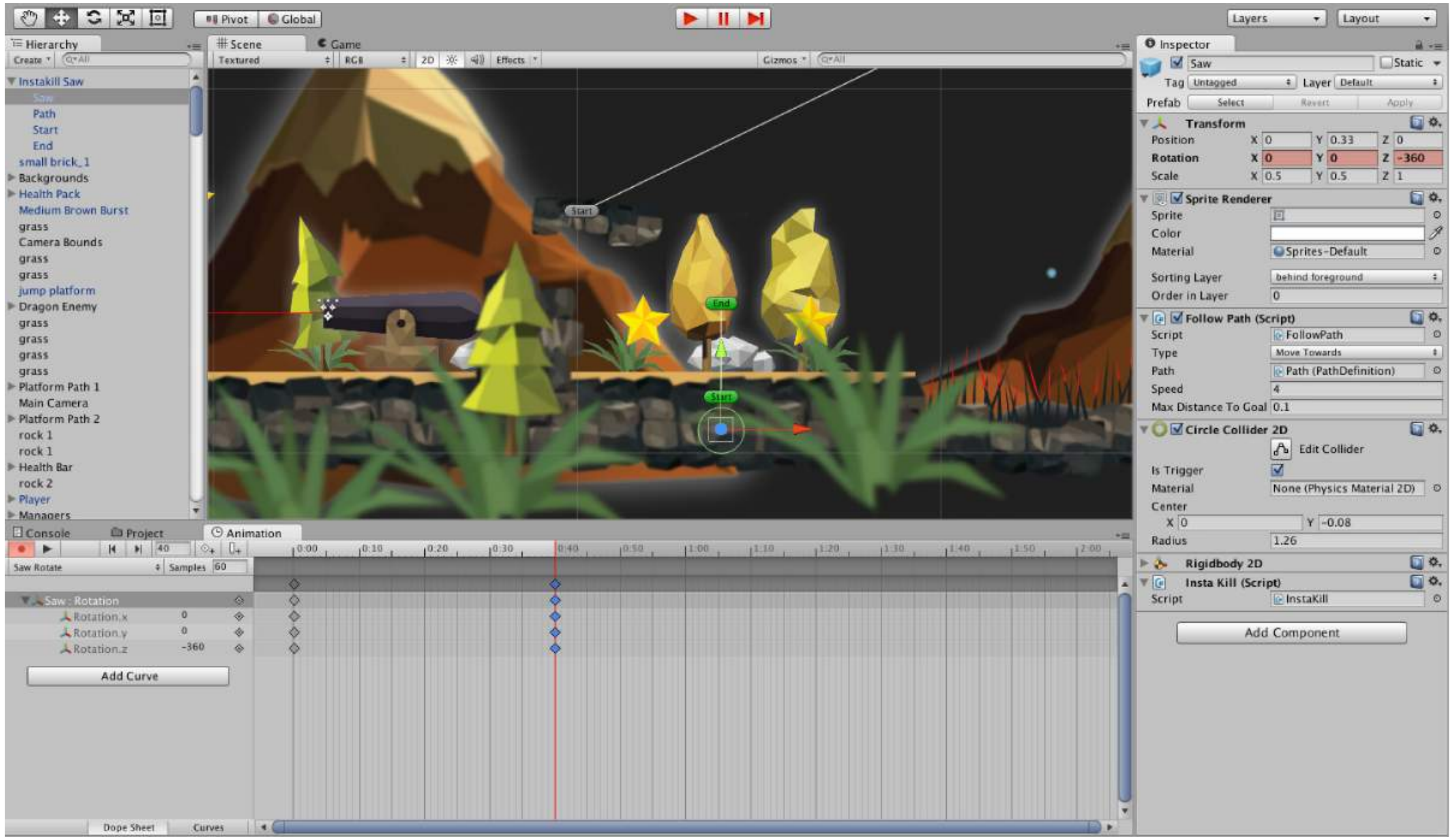


jump platform Static
Tag: Untagged Layer: Platforms
Prefab: Select Revert Apply
Sprite: jump platform
Material: Sprites-Default
Sorting Layer: Default
Order in Layer: 0
Script: JumpPlatform
Jump Magnitude: 20
Jump Sound: Bounce-SoundBible.com-

- ANIMATIONS (BASIC)

I think by having objects animated would really make the whole gaming experience more immersive by having less static objects in the game world as it feels more like a living/ breathing world.

For the saw, I created a basic rotation animation for its prefab, meaning this animation would be set to all saw prefabbed objects in the scene. I set z-axis to -360 in under half a second so would look like it was spinning out of the ground.



Saw Static
Tag Untagged Layer Default

Transform
Position X 0 Y 0.33 Z 0
Rotation X 0 Y 0 Z -360
Scale X 0.5 Y 0.5 Z 1

Sprite Renderer
Sprite
Color
Material Sprites-Default
Sorting Layer behind foreground
Order in Layer 0

Follow Path (Script)
Script FollowPath
Type Move Towards
Path Path (PathDefinition)
Speed 4
Max Distance To Goal 0.1

Circle Collider 2D
Is Trigger ✓
Material None (Physics Material 2D)
Center X 0 Y -0.08
Radius 1.26

Rigidbody 2D

Insta Kill (Script)
Script InstaKill

Saw Rotate Samples 60
Saw : Rotation
Rotation.x 0
Rotation.y 0
Rotation.z -360

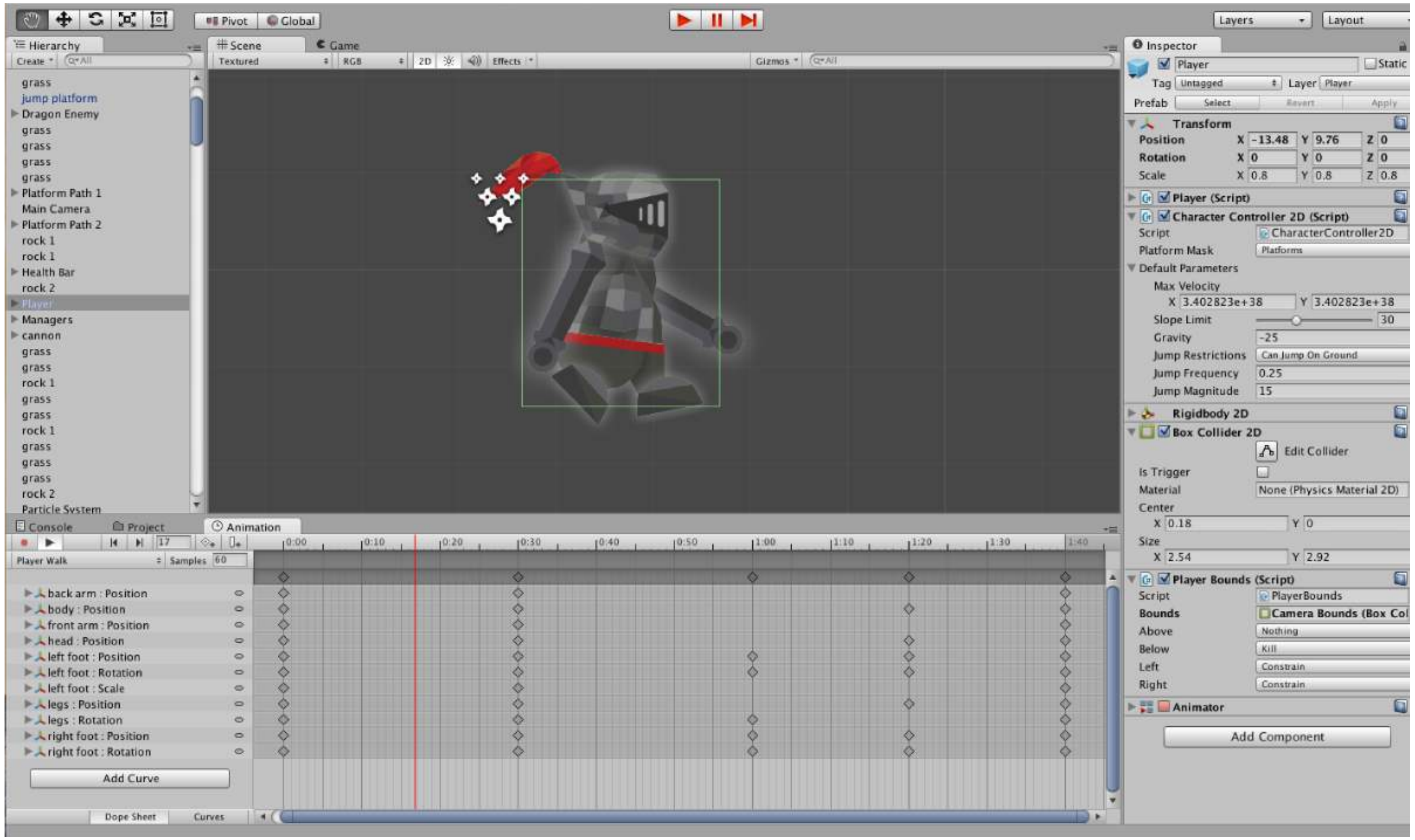
GAME DEVELOPMENT

- ANIMATIONS (PLAYER)

I want my player to be animated so he feels more realistic and less static.

These animations will include and idle animation, so he looks like he is breathing. A walk animation, a jump animation, a fire projectile animation and finally a death animation.

To animate, it is very simple. Similar to the basic animations before hand I move the body parts accordingly so it looks like they are walking, jumping etc.



Hierarchy: grass, jump platform, Dragon Enemy, grass, grass, grass, grass, Platform Path 1, Main Camera, Platform Path 2, rock 1, rock 1, Health Bar, rock 2, Player, Managers, cannon, grass, grass, rock 1, grass, grass, rock 1, grass, grass, grass, rock 2, Particle System

Animation: Player Walk | Samples 60 | 17
0:00 0:10 0:20 0:30 0:40 0:50 1:00 1:10 1:20 1:30 1:40

back arm : Position
body : Position
front arm : Position
head : Position
left foot : Position
left foot : Rotation
left foot : Scale
legs : Position
legs : Rotation
right foot : Position
right foot : Rotation

Inspector:

Player			Static
Tag	Untagged	Layer	Player
Prefab	Select	Revert	Apply

Transform

Position	X -13.48	Y 9.76	Z 0
Rotation	X 0	Y 0	Z 0
Scale	X 0.8	Y 0.8	Z 0.8

Player (Script)

Character Controller 2D (Script)

Script	CharacterController2D
Platform Mask	Platforms
Default Parameters	
Max Velocity	X 3.402823e+38 Y 3.402823e+38
Slope Limit	30
Gravity	-25
Jump Restrictions	Can Jump On Ground
Jump Frequency	0.25
Jump Magnitude	15

Rigidbody 2D

Box Collider 2D

Is Trigger	☐
Material	None (Physics Material 2D)
Center	X 0.18 Y 0
Size	X 2.54 Y 2.92

Player Bounds (Script)

Script	PlayerBounds
Bounds	Camera Bounds (Box Col
Above	Nothing
Below	Kill
Left	Constrain
Right	Constrain

Animator

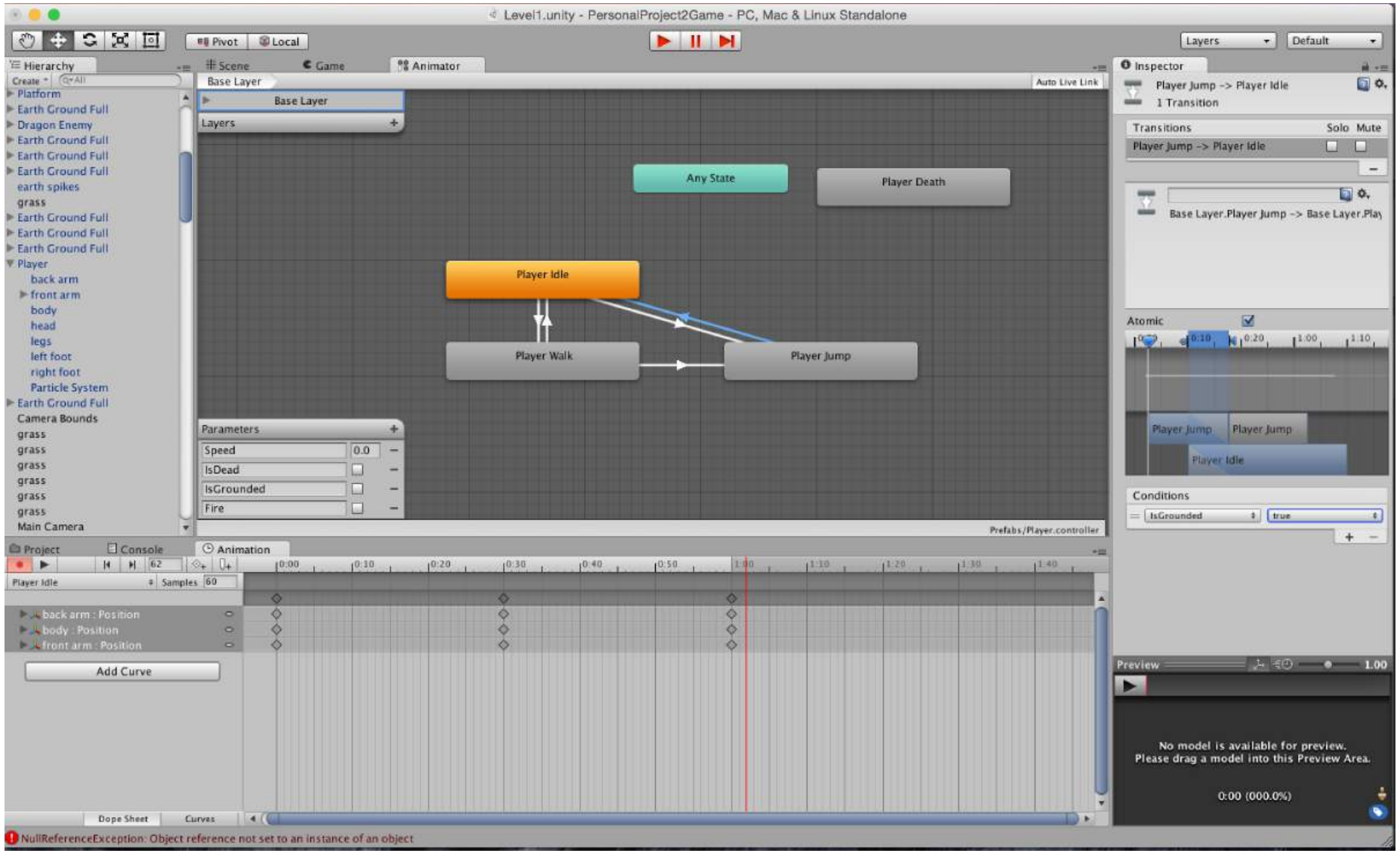
GAME DEVELOPMENT

- ANIMATION CONTROLLER

After I had animated my character in the different states, it is now time to control those animations according to what the player inputs from the controller.

In the animation controller, I implemented all of the player states and I needed to create transitions between these states.

Parameters take a big part of these transitions as they decide what state to play when the player is doing something. These transitions are conversed so the animations will always go back to the idle animation.



NullReferenceException: Object reference not set to an instance of an object

GAME DEVELOPMENT

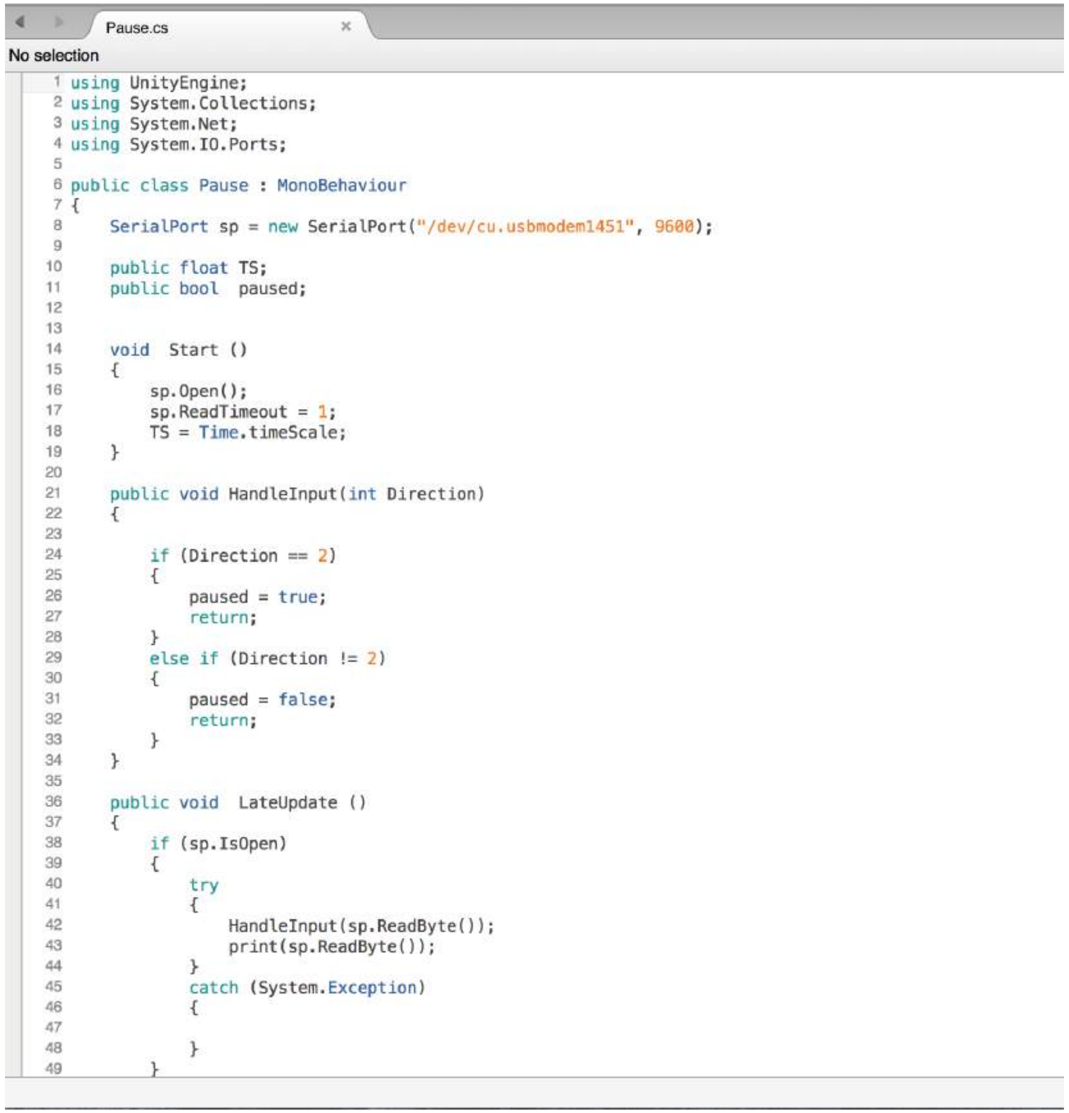
- PAUSING THE GAME

I think my game needs some sort of pause option built in; I want to make a unique pause effect though, not just a simple "hit a button and everything pauses suddenly" type of effect.

I created a new C# script and read from the USB port that my Arduino is connected to. "2" is identified as the ring finger, so when the ring finger are connected, this would be the action for pause.

Note - This is linked to the glove controller test that reads from the USB port, go to page 144 for this information.

(Part 1 of code)



```
using UnityEngine;
using System.Collections;
using System.Net;
using System.IO.Ports;

public class Pause : MonoBehaviour
{
    SerialPort sp = new SerialPort("/dev/cu.usbmodem1451", 9600);

    public float TS;
    public bool  paused;


    void  Start ()
    {
        sp.Open();
        sp.ReadTimeout = 1;
        TS = Time.timeScale;
    }

    public void HandleInput(int Direction)
    {

        if (Direction == 2)
        {
            paused = true;
            return;
        }
        else if (Direction != 2)
        {
            paused = false;
            return;
        }
    }

    public void  LateUpdate ()
    {
        if (sp.IsOpen)
        {
            try
            {
                HandleInput(sp.ReadByte());
                print(sp.ReadByte());
            }
            catch (System.Exception)
            {

            }
        }
```

GAME DEVELOPMENT

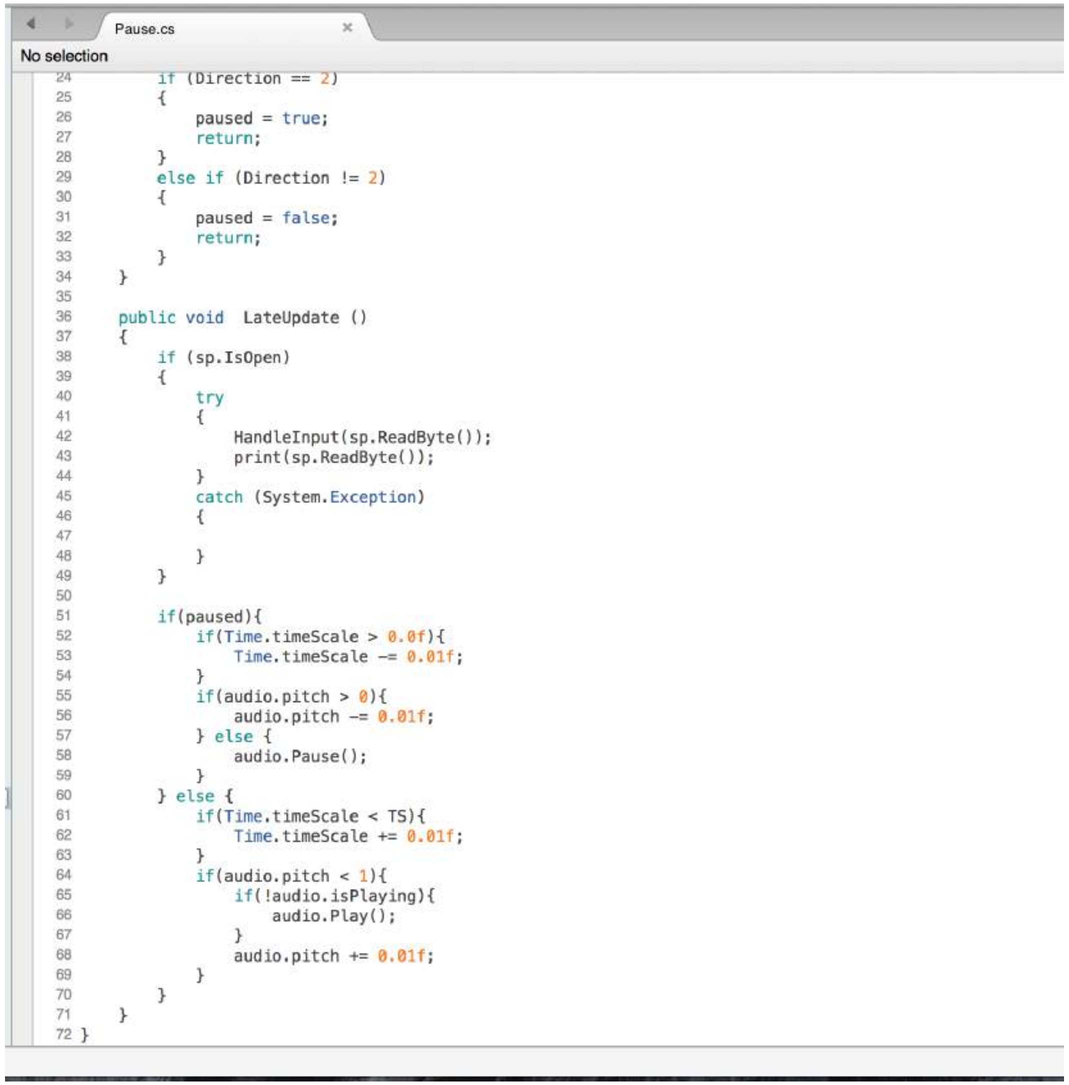
- PAUSING THE GAME

(Part 2 of code)

The end part of this code is where the actual pause effect is created.

This pause effect is slow motion, when a connection is made, whatever that is moving in the game world will slowly come to a stop, including the music.

By editing the "Time. timeScale" this is possible, if unpaused the time scale is normal, but when it is pressed everything will slow right down until nothing is happening, but then when a connection is not made, it all returns back to normal speed.

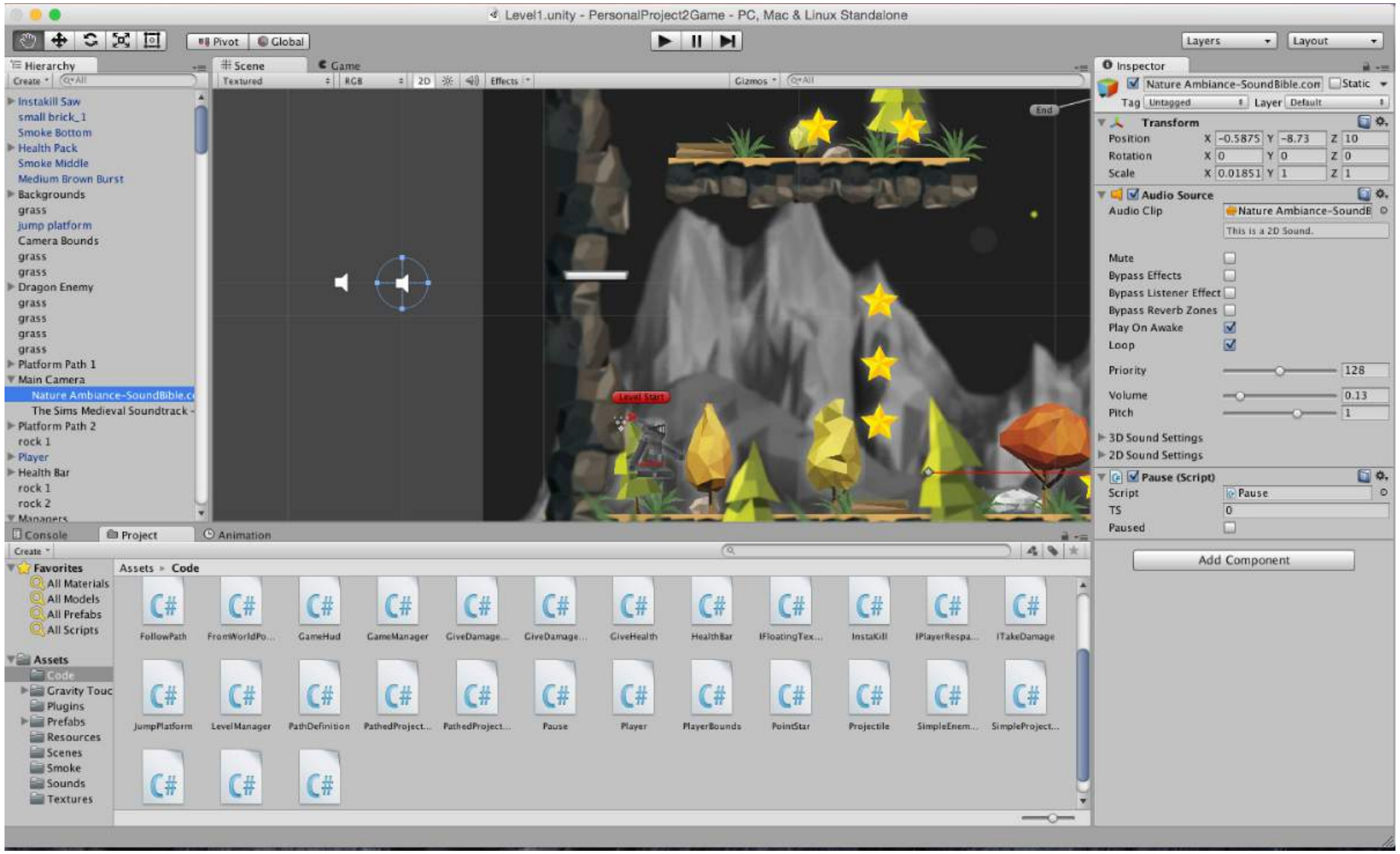


```
        if (Direction == 2)
        {
            paused = true;
            return;
        }
        else if (Direction != 2)
        {
            paused = false;
            return;
        }
    }

    public void LateUpdate ()
    {
        if (sp.IsOpen)
        {
            try
            {
                HandleInput(sp.ReadByte());
                print(sp.ReadByte());
            }
            catch (System.Exception)
            {

            }
        }

        if(paused){
            if(Time.timeScale > 0.0f){
                Time.timeScale -= 0.01f;
            }
            if(audio.pitch > 0){
                audio.pitch -= 0.01f;
            } else {
                audio.Pause();
            }
        } else {
            if(Time.timeScale < TS){
                Time.timeScale += 0.01f;
            }
            if(audio.pitch < 1){
                if(!audio.isPlaying){
                    audio.Play();
                }
                audio.pitch += 0.01f;
            }
        }
    }
}
```

- PAUSING THE GAME

Back in Unity, I have to assign this script to both of my backing tracks in order for it to work. Then i press play and everything is working as it should.

Note - you may notice a slight graphical change in the game, the development of this can be seen from page 157 about further development of the graphics.


Level1.unity - PersonalProject2Game - PC, Mac & Linux Standalone

Hierarchy: Instakill Saw, small brick_1, Smoke Bottom, Health Pack, Smoke Middle, Medium Brown Burst, Backgrounds, grass, jump platform, Camera Bounds, grass, grass, Dragon Enemy, grass, grass, grass, grass, Platform Path 1, Main Camera, Nature Ambiance-SoundBible.c, The Sims Medieval Soundtrack -, Platform Path 2, rock 1, Player, Health Bar, rock 1, rock 2, Managers

Inspector: Nature Ambiance-SoundBible.com — Tag Untagged — Layer Default

Transform — Position X -0.5875 Y -8.73 Z 10; Rotation X 0 Y 0 Z 0; Scale X 0.01851 Y 1 Z 1

Audio Source — Audio Clip: Nature Ambiance-SoundB (This is a 2D Sound.); Mute ☐; Bypass Effects ☐; Bypass Listener Effect ☐; Bypass Reverb Zones ☐; Play On Awake ✓; Loop ✓; Priority 128; Volume 0.13; Pitch 1; 3D Sound Settings; 2D Sound Settings

Pause (Script) — Script: Pause; TS 0; Paused ☐

Add Component

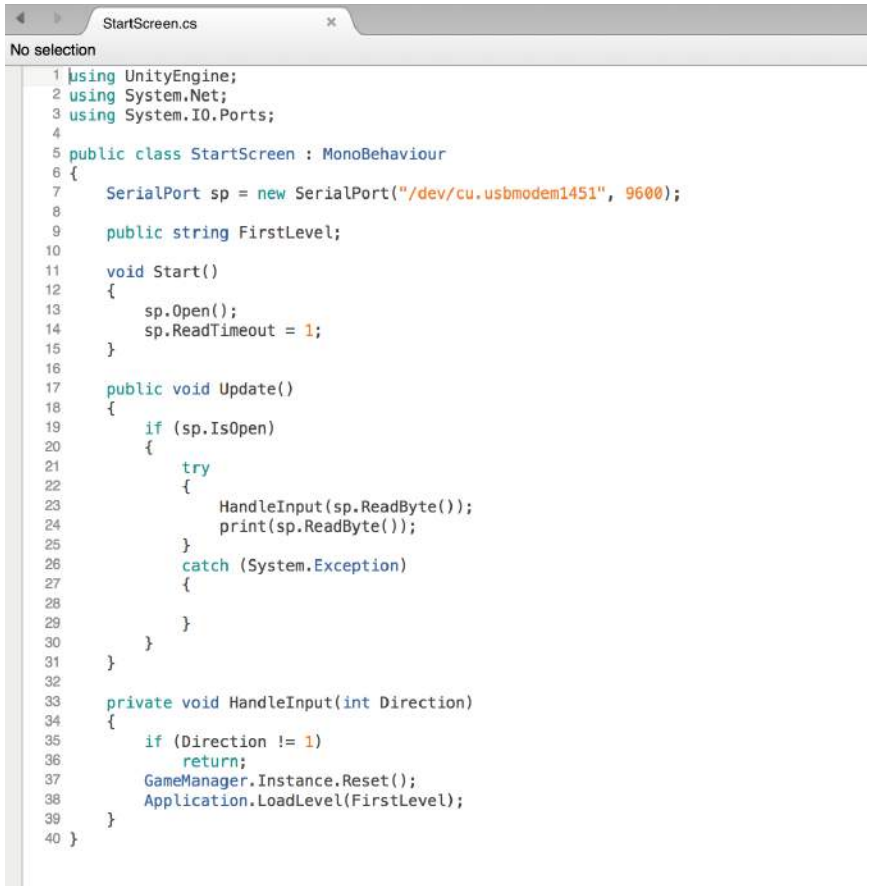
Project: Assets ▸ Code — FollowPath, FromWorldPo..., GameHud, GameManager, GiveDamage..., GiveDamage..., GiveHealth, HealthBar, IFloatingTex..., InstaKill, IPlayerRespa..., ITakeDamage, JumpPlatform, LevelManager, PathDefinition, PathedProject..., PathedProject..., Pause, Player, PlayerBounds, PointStar, Projectile, SimpleEnem..., SimpleProject...

Favorites: All Materials, All Models, All Prefabs, All Scripts. Assets: Code, Gravity Touc, Plugins, Prefabs, Resources, Scenes, Smoke, Sounds, Textures

- MULTIPLE LEVELS

My game needs a start screen; I create a new .cs script that will be active as soon as the game is played. What this basically does is when an input is performed; the game will go to a new scene, level 1.

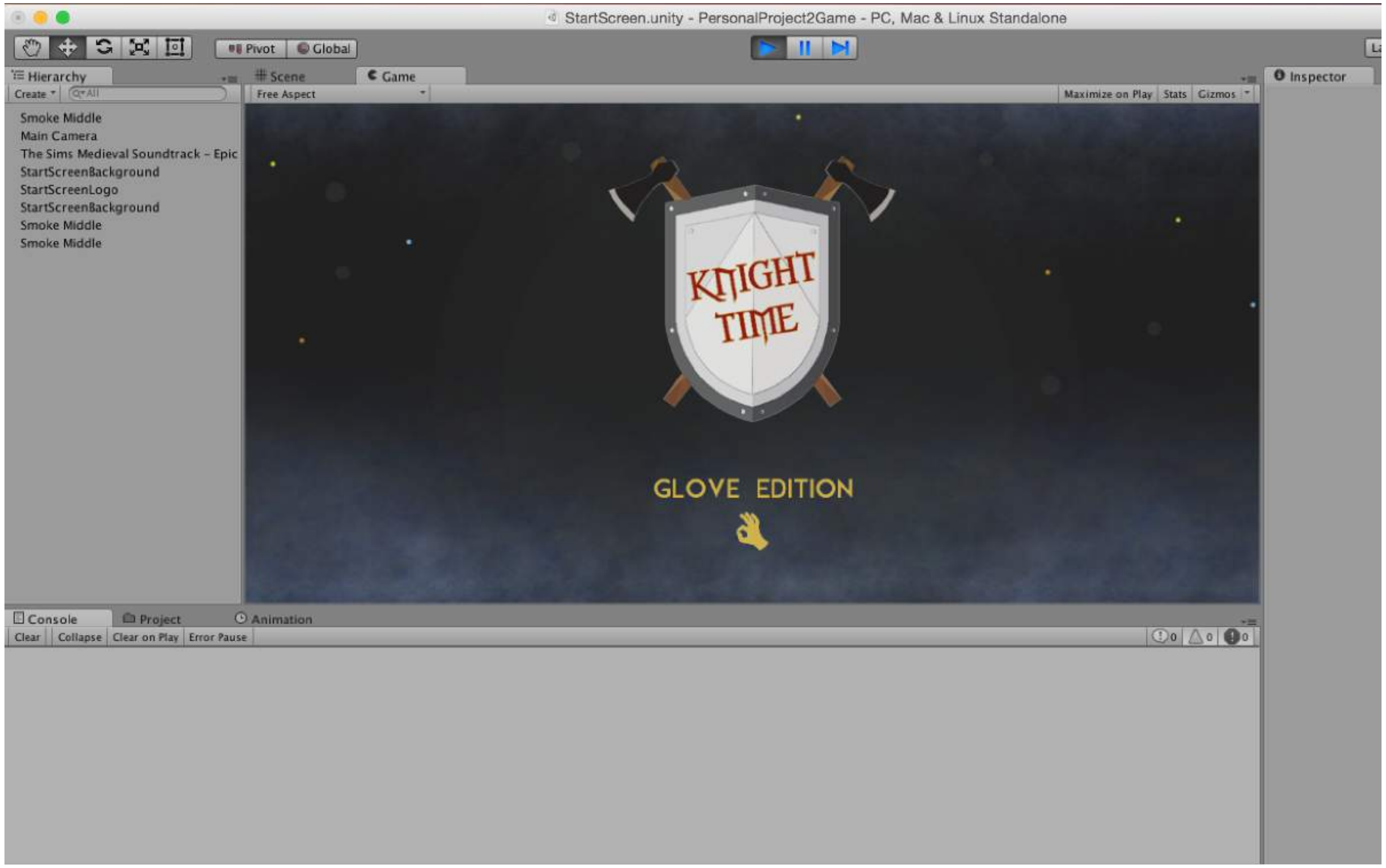
Note - This is linked to the glove controller test that reads from the USB port, go to page 144 for this information.



```
using UnityEngine;
using System.Net;
using System.IO.Ports;

public class StartScreen : MonoBehaviour
{
    SerialPort sp = new SerialPort("/dev/cu.usbmodem1451", 9600);

    public string FirstLevel;

    void Start()
    {
        sp.Open();
        sp.ReadTimeout = 1;
    }

    public void Update()
    {
        if (sp.IsOpen)
        {
            try
            {
                HandleInput(sp.ReadByte());
                print(sp.ReadByte());
            }
            catch (System.Exception)
            {

            }
        }
    }

    private void HandleInput(int Direction)
    {
        if (Direction != 1)
            return;
        GameManager.Instance.Reset();
        Application.LoadLevel(FirstLevel);
    }
}
```

- MULTIPLE LEVELS

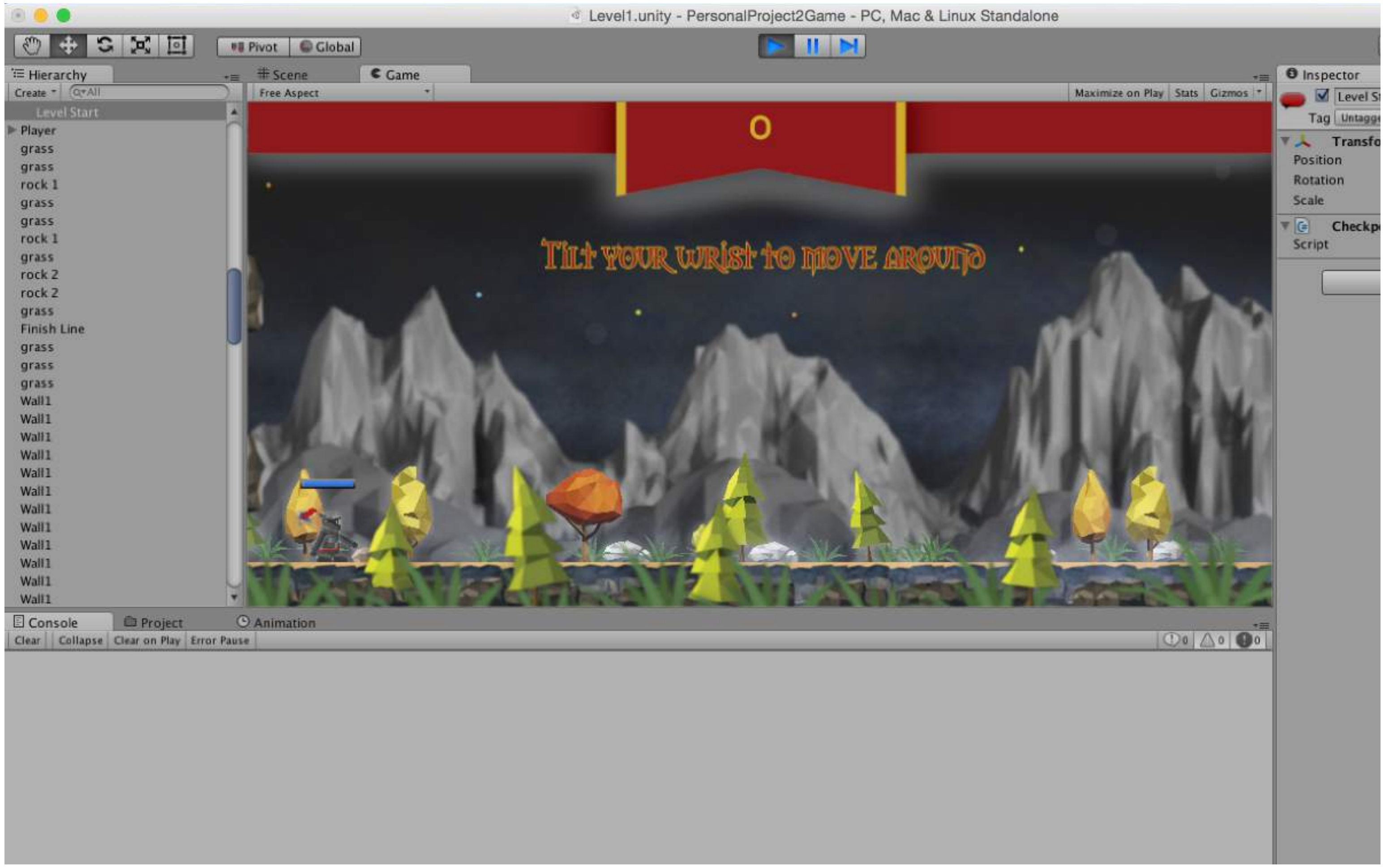
In Unity I create a new scene that has nothing in it, then add the logo (Please see page 165 for the logo development) and the a smoke particle effect system to add visual effect. As well as the "StartScreen.cs" script that has just been written to the main camera.



- MULTIPLE LEVELS

Level 1 shows off my new aesthetics well, I opted for a type of "Boot camp" level where it would take the user through the controls and get to grips with how the game plays. I think this is an important level as it is a starting point, it tells the user what to do.

There is not a lot of challenge in this level but it is a very good demonstrator for Knight Time's first level. (Please see version 5 of my prototype on page 157 for more details about the graphical change).

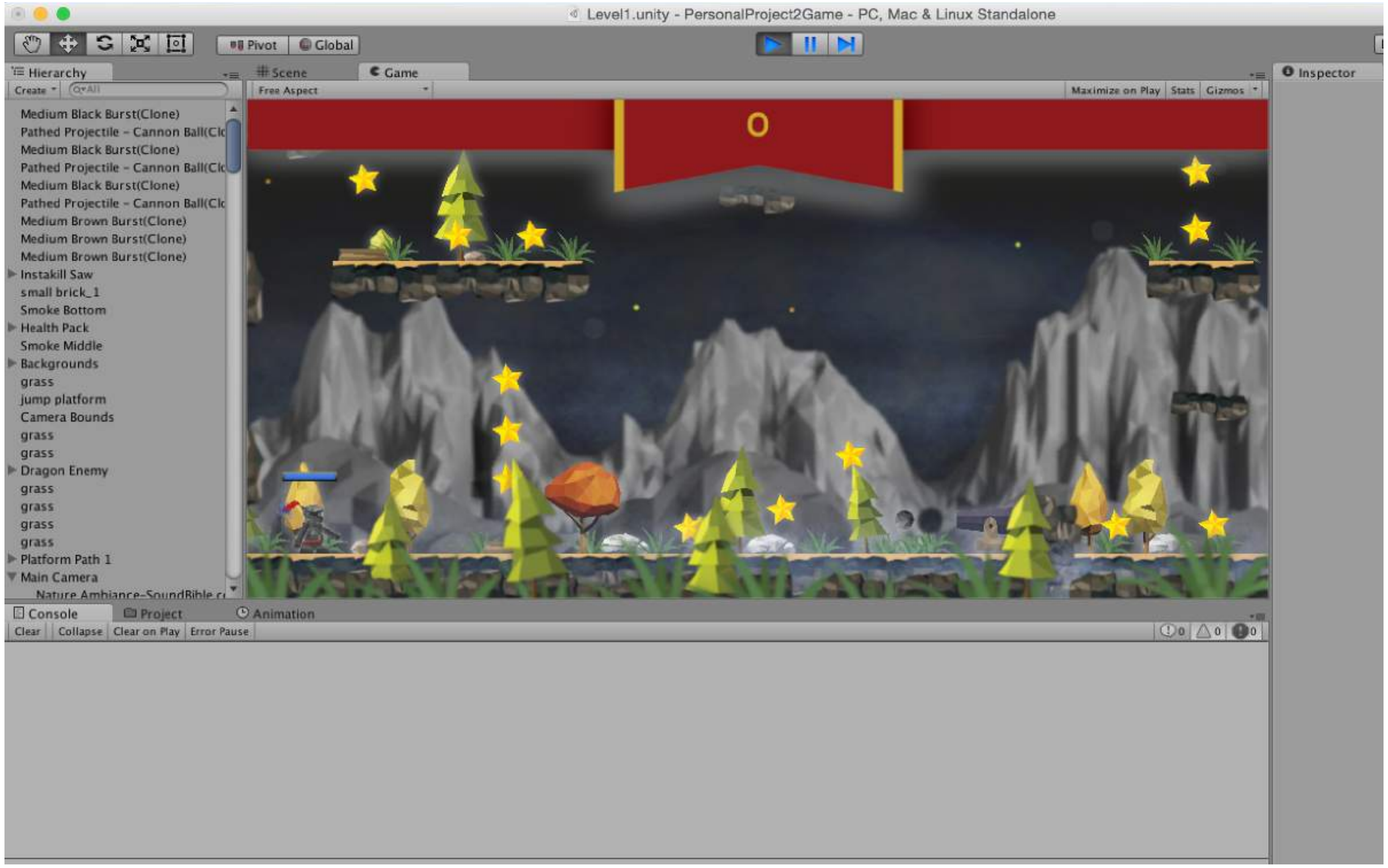


GAME DEVELOPMENT

- MULTIPLE LEVELS

Level 2 was added in as the main level for the game. It is the most challenging level yet,

The point of this level was to include enough elements such as traps and enemies so the user could exercise their hands enough without getting in a tangle and a tired hand. The level can last to anywhere around a minute and I feel this is a sufficient amount of time for the hand exercises to be performed per level.



Level1.unity - PersonalProject2Game - PC, Mac & Linux Standalone
Pivot
Global
Hierarchy
Create
Scene
Game
Free Aspect
Maximize on Play
Stats
Gizmos
Inspector
Medium Black Burst(Clone)
Pathed Projectile - Cannon Ball(Cl
Medium Black Burst(Clone)
Pathed Projectile - Cannon Ball(Cl
Medium Black Burst(Clone)
Pathed Projectile - Cannon Ball(Cl
Medium Brown Burst(Clone)
Medium Brown Burst(Clone)
Medium Brown Burst(Clone)
Instakill Saw
small brick_1
Smoke Bottom
Health Pack
Smoke Middle
Backgrounds
grass
jump platform
Camera Bounds
grass
grass
Dragon Enemy
grass
grass
grass
grass
Platform Path 1
Main Camera
Nature Ambiance-SoundBible.c
0
Console
Project
Animation
Clear
Collapse
Clear on Play
Error Pause
0
0
0

GAME DEVELOPMENT

- CONCLUSION OF SECTION

Now that my game is completed to a standard in which I am very happy with.

It has been a very long and sometimes very hard process to code in C# but I feel like what I have done has taught me this coding language enough so I know how games are built in the back end, its not all about good graphics.

I have had feedback throughout and this has helped greatly in getting my game built with features that people want, such as a dragon enemy and checkpoints.

I think 2 levels are a substantial amount for what I am trying to prove as yes I've built a game with a storyline but I don't need to build a full game with 100's of levels.

This Demo, I feel shows off enough to demonstrate my skills within Unity and C#.

DESIGN DEVELOPMENT

Now lets make the glove

GLOVE CONTROLLER

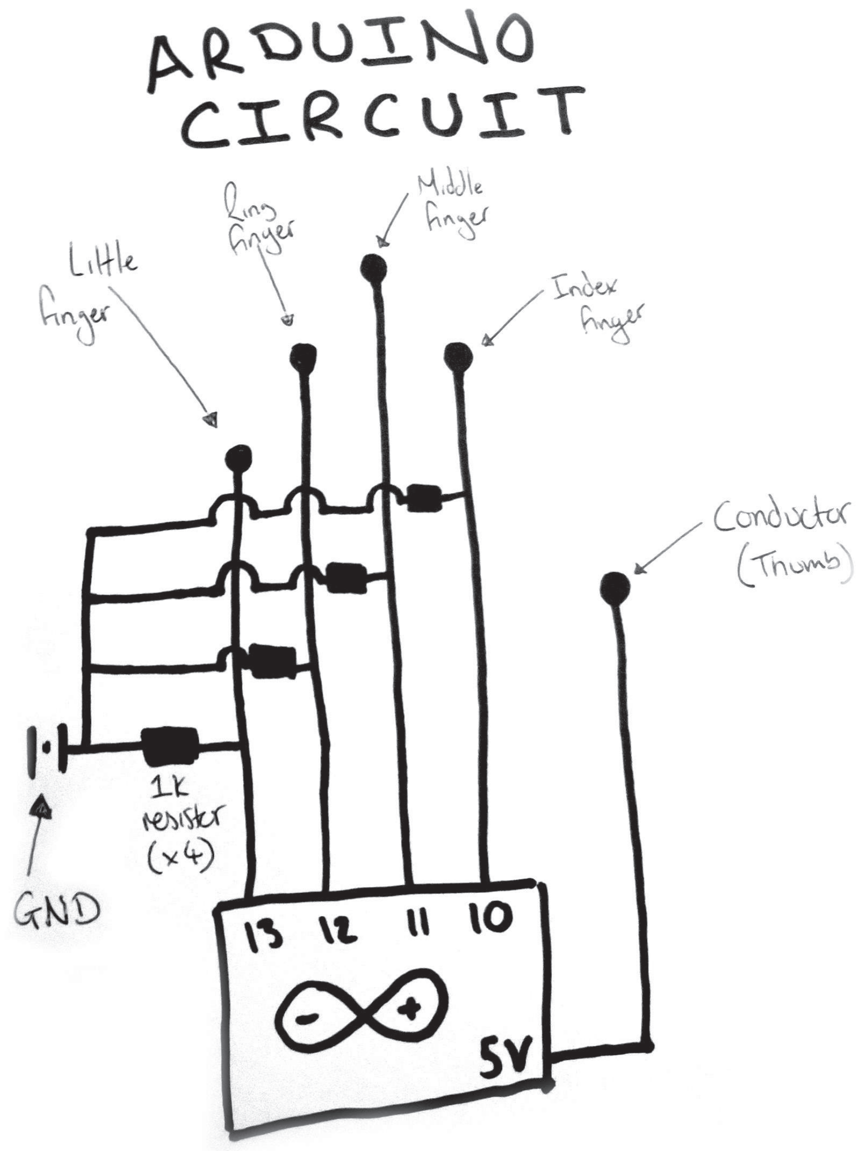
Now that I have my game built to a standard that I am happy to show off, I will focus on the actual glove controller for the interaction side, I want each gesture to represent the muscle flex that can help with arthritis.

I want these gestures to be easy to perform by any user so from my research I will have to define my controls and also make this possible in Arduino then link it up to Unity through the serial port read, hopefully this will work.

GLOVE CONTROLLER

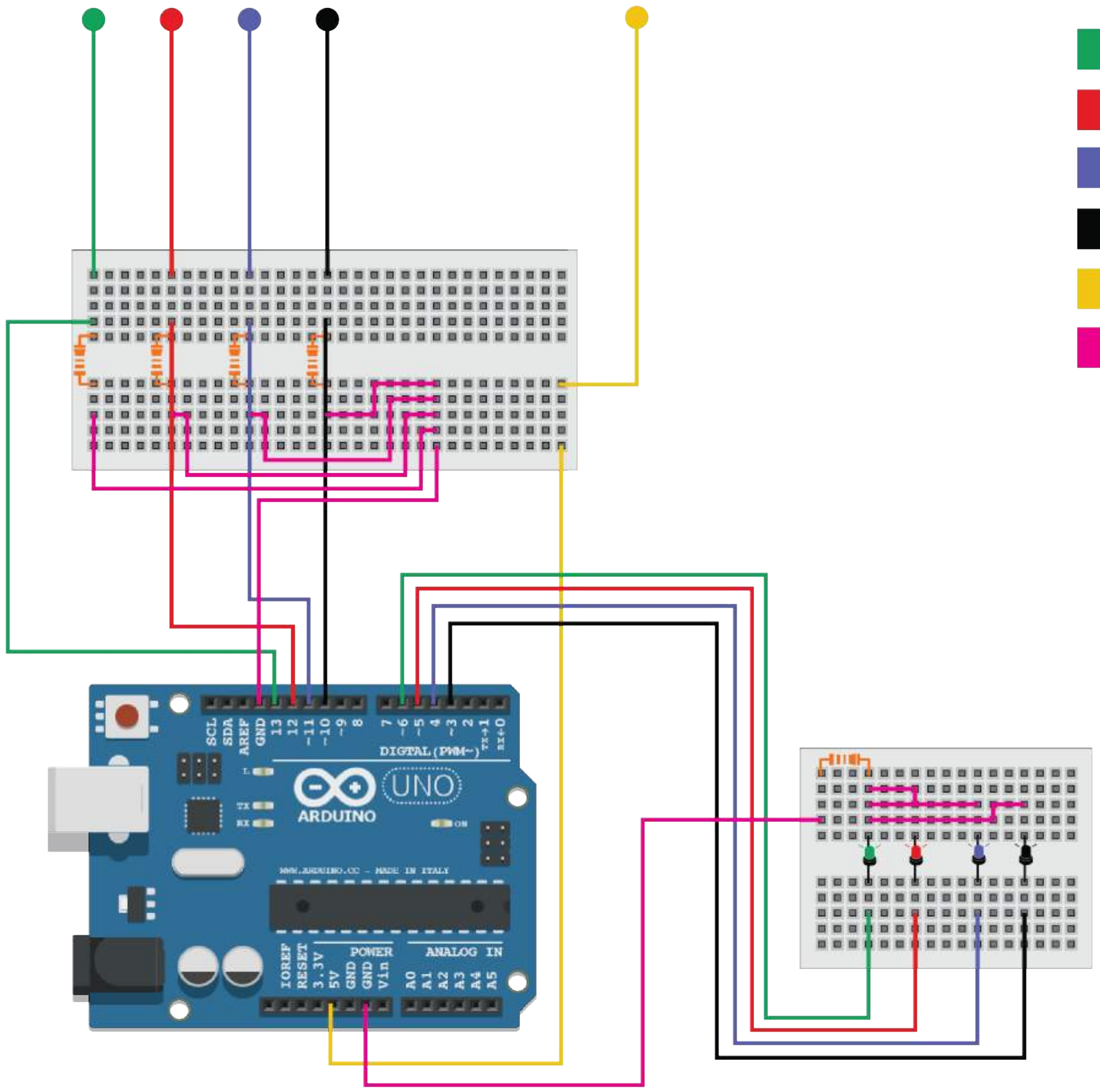
With a little help from Ben, we were able to sketch out a basic diagram of how the glove controller would work as a circuit.

The diagram is sketched out to look like a skeleton, basically imagine it as a broken circuit, the thumb is connected to 5v and the fingers are grounded. Once the thumb has connected with any of the fingers, a connection is made, this will be made by the 'thumb touch' exercise that I outlined from my research and this connection will give a value, this value is what I will use to control the player character.



GLOVE CONTROLLER

This diagram is neatened up from my sketch with the added illustration of LED's, this will help me identify when a connection is made by the thumb and a finger as the corresponding LED will light up.

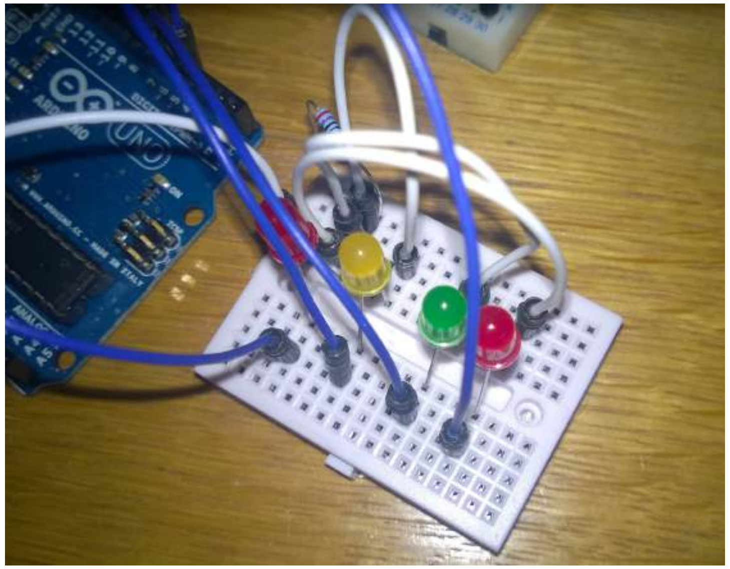


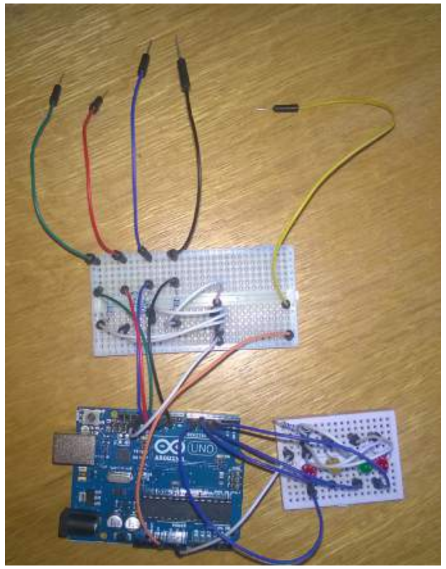
KEY:

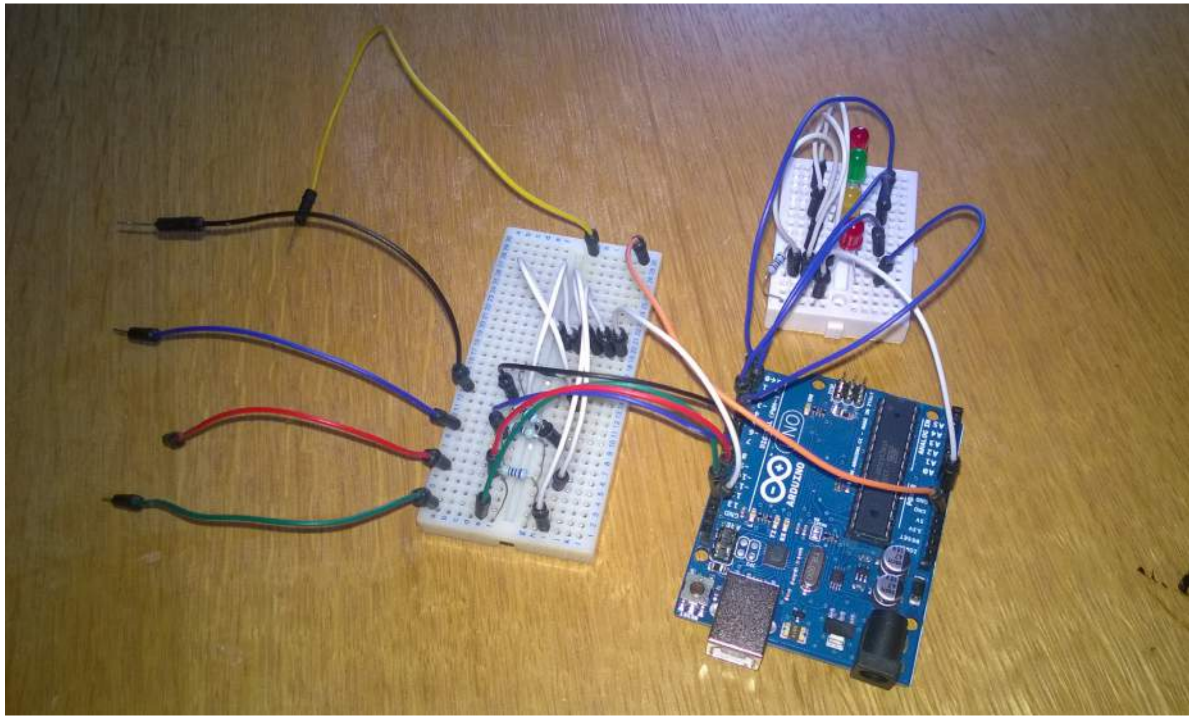
🟩	LITTLE FINGER
🟥	RING FINGER
🟦	MIDDLE FINGER
⬛	INDEX FINGER
🟨	THUMB
🟪	GND

GLOVE CONTROLLER

This is my initial glove controller concept, built from the diagram with everything connected together. Obviously I will extend the wires at a later date as they are not long enough but as a starting point this should work.





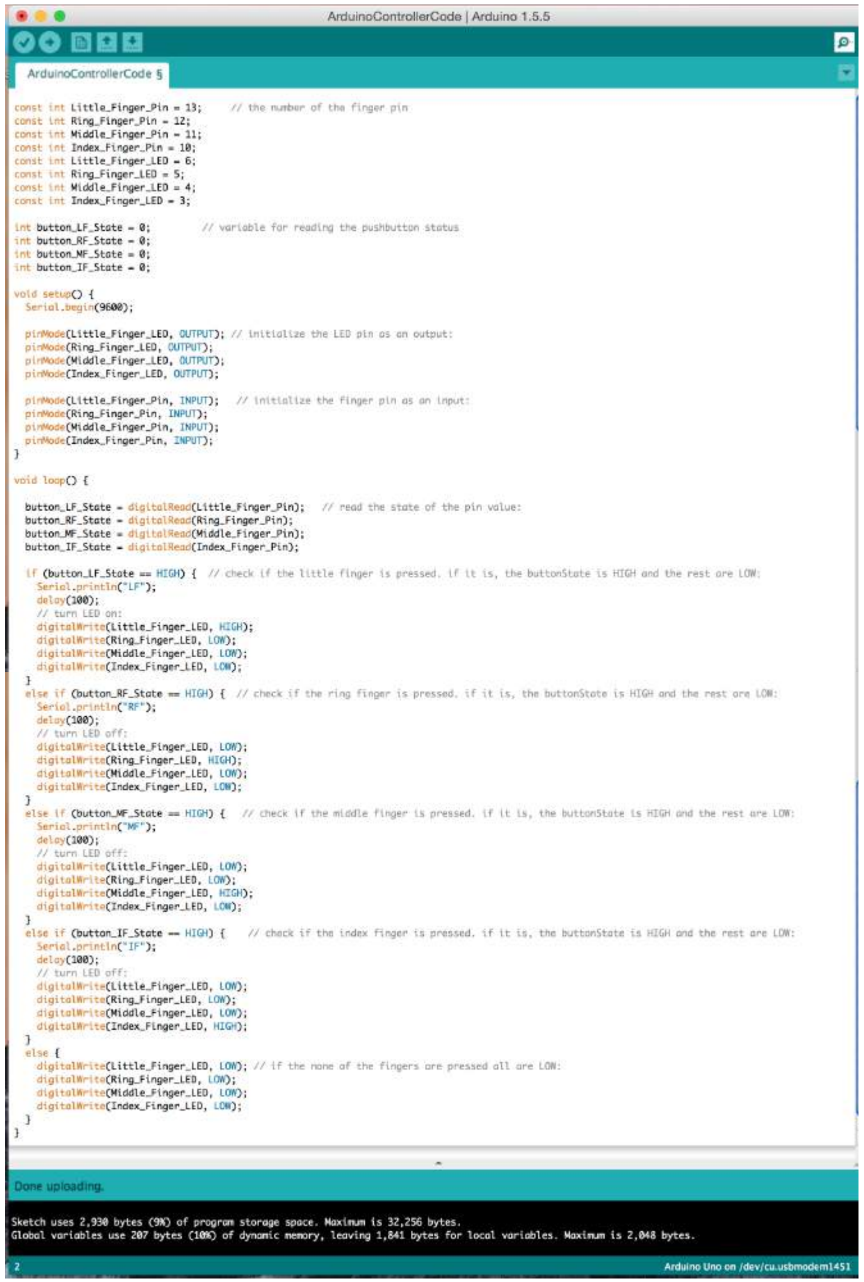


GLOVE CONTROLLER

For my Arduino code to do what I want it to, basically when a finger and thumb it's connected it should give out a value in the serial monitor, then this value will be what I will implement in Unity at a later date.

However, my initial set up of the Arduino code says that when a connected is made by one finger, it lights up a corresponding LED and gives out a value, e.g. "LF" for little finger while the other 3 fingers are inactive.

When no connections are made, everything is inactive.



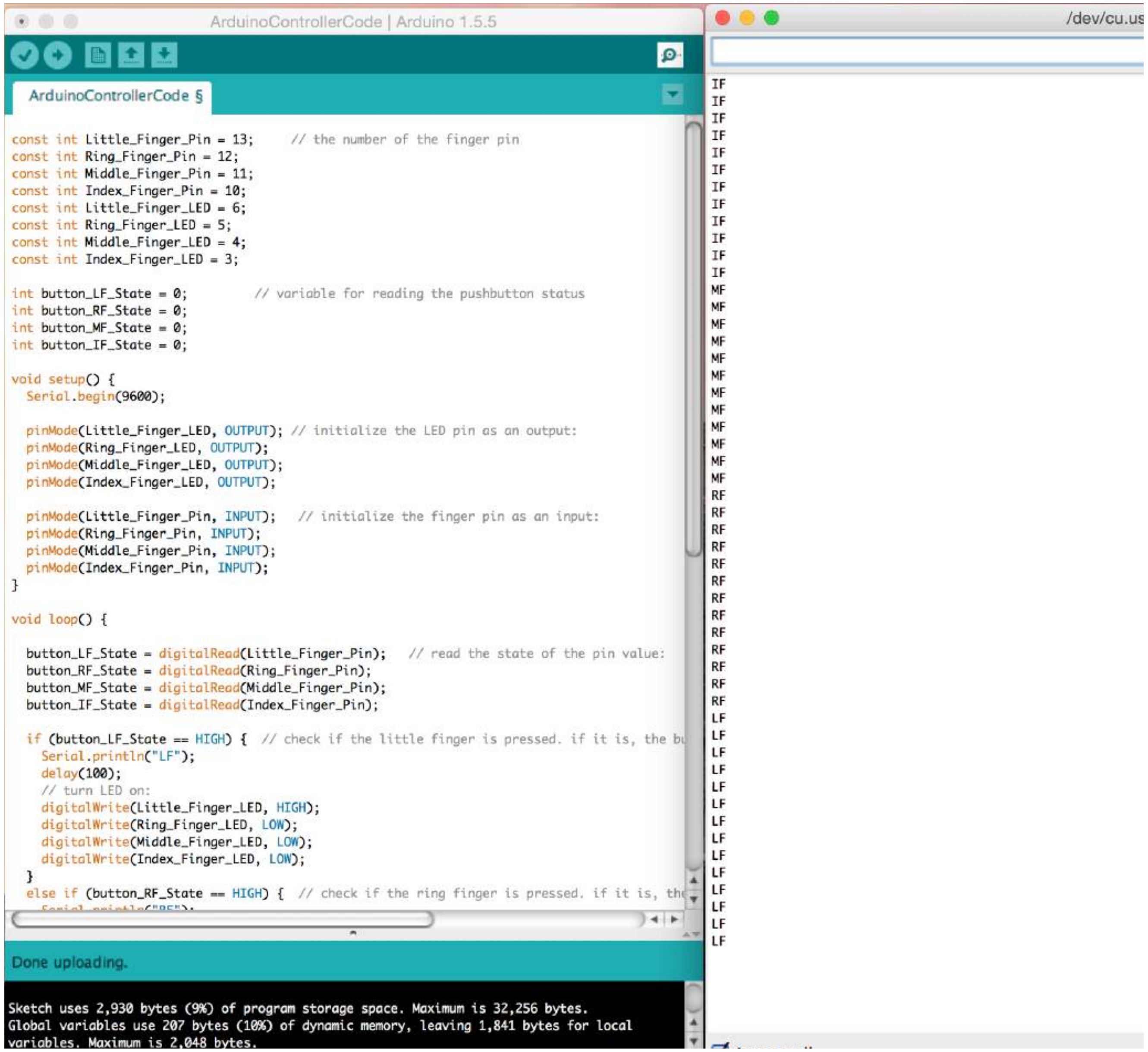
```
const int Little_Finger_Pin = 13;     // the number of the finger pin
const int Ring_Finger_Pin = 12;
const int Middle_Finger_Pin = 11;
const int Index_Finger_Pin = 10;
const int Little_Finger_LED = 6;
const int Ring_Finger_LED = 5;
const int Middle_Finger_LED = 4;
const int Index_Finger_LED = 3;

int button_LF_State = 0;         // variable for reading the pushbutton status
int button_RF_State = 0;
int button_MF_State = 0;
int button_IF_State = 0;

void setup() {
  Serial.begin(9600);

  pinMode(Little_Finger_LED, OUTPUT); // initialize the LED pin as an output:
  pinMode(Ring_Finger_LED, OUTPUT);
  pinMode(Middle_Finger_LED, OUTPUT);
  pinMode(Index_Finger_LED, OUTPUT);

  pinMode(Little_Finger_Pin, INPUT);   // initialize the finger pin as an input:
  pinMode(Ring_Finger_Pin, INPUT);
  pinMode(Middle_Finger_Pin, INPUT);
  pinMode(Index_Finger_Pin, INPUT);
}

void loop() {

  button_LF_State = digitalRead(Little_Finger_Pin);   // read the state of the pin value:
  button_RF_State = digitalRead(Ring_Finger_Pin);
  button_MF_State = digitalRead(Middle_Finger_Pin);
  button_IF_State = digitalRead(Index_Finger_Pin);

  if (button_LF_State == HIGH) {  // check if the little finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.println("LF");
    delay(100);
    // turn LED on:
    digitalWrite(Little_Finger_LED, HIGH);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_RF_State == HIGH) {  // check if the ring finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.println("RF");
    delay(100);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, HIGH);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_MF_State == HIGH) {   // check if the middle finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.println("MF");
    delay(100);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, HIGH);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_IF_State == HIGH) {    // check if the index finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.println("IF");
    delay(100);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, HIGH);
  }
  else {
    digitalWrite(Little_Finger_LED, LOW); // if the none of the fingers are pressed all are LOW:
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
}
```

```
Done uploading.

Sketch uses 2,930 bytes (9%) of program storage space. Maximum is 32,256 bytes.
Global variables use 207 bytes (10%) of dynamic memory, leaving 1,841 bytes for local variables. Maximum is 2,048 bytes.
```

GLOVE CONTROLLER

Once the code has uploaded to my Arduino Uno and I opened the serial monitor, the serial print is giving off the correct values when a correct connection is made.

These values "LF, RF, MF, IF" represent the different finger movements and the controller buttons. There is only one thing wrong with this method, Unity only understands numbers and not letters, so instead of the serial print being, for example, "LF", it would be "1" and the other fingers would correspond with "2", "3" and "4" so the controller would work correctly.



```
const int Little_Finger_Pin = 13;     // the number of the finger pin
const int Ring_Finger_Pin = 12;
const int Middle_Finger_Pin = 11;
const int Index_Finger_Pin = 10;
const int Little_Finger_LED = 6;
const int Ring_Finger_LED = 5;
const int Middle_Finger_LED = 4;
const int Index_Finger_LED = 3;

int button_LF_State = 0;         // variable for reading the pushbutton status
int button_RF_State = 0;
int button_MF_State = 0;
int button_IF_State = 0;

void setup() {
  Serial.begin(9600);

  pinMode(Little_Finger_LED, OUTPUT); // initialize the LED pin as an output:
  pinMode(Ring_Finger_LED, OUTPUT);
  pinMode(Middle_Finger_LED, OUTPUT);
  pinMode(Index_Finger_LED, OUTPUT);

  pinMode(Little_Finger_Pin, INPUT);   // initialize the finger pin as an input:
  pinMode(Ring_Finger_Pin, INPUT);
  pinMode(Middle_Finger_Pin, INPUT);
  pinMode(Index_Finger_Pin, INPUT);
}

void loop() {

  button_LF_State = digitalRead(Little_Finger_Pin);   // read the state of the pin value:
  button_RF_State = digitalRead(Ring_Finger_Pin);
  button_MF_State = digitalRead(Middle_Finger_Pin);
  button_IF_State = digitalRead(Index_Finger_Pin);

  if (button_LF_State == HIGH) {  // check if the little finger is pressed. if it is, the bu
    Serial.println("LF");
    delay(100);
    // turn LED on:
    digitalWrite(Little_Finger_LED, HIGH);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_RF_State == HIGH) {  // check if the ring finger is pressed. if it is, the
    Serial.println("RF");
```

Done uploading.

```
Sketch uses 2,930 bytes (9%) of program storage space. Maximum is 32,256 bytes.
Global variables use 207 bytes (10%) of dynamic memory, leaving 1,841 bytes for local
variables. Maximum is 2,048 bytes.
```

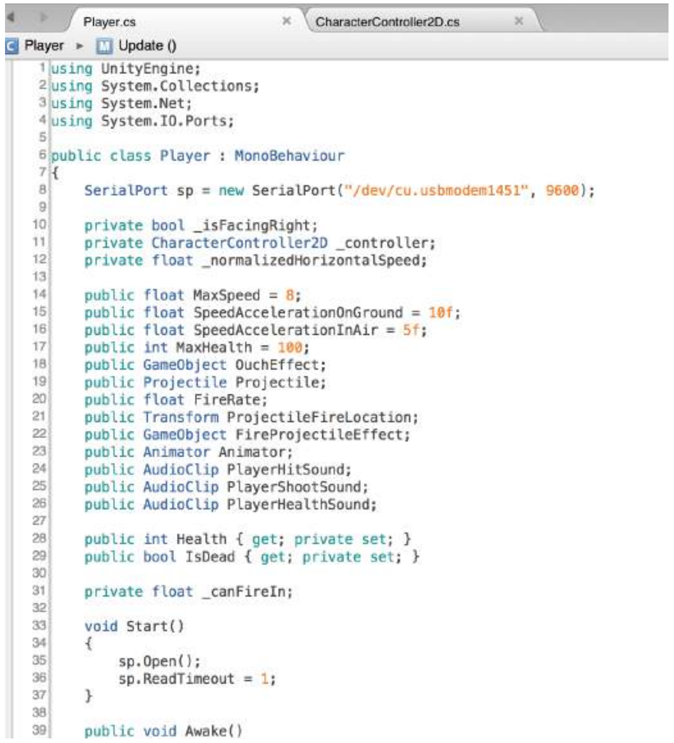
```
IF
IF
IF
IF
IF
IF
IF
IF
IF
IF
IF
IF
MF
MF
MF
MF
MF
MF
MF
MF
MF
MF
MF
MF
RF
RF
RF
RF
RF
RF
RF
RF
RF
RF
RF
RF
RF
LF
LF
LF
LF
LF
LF
LF
LF
LF
LF
LF
LF
LF
LF
```

GLOVE CONTROLLER

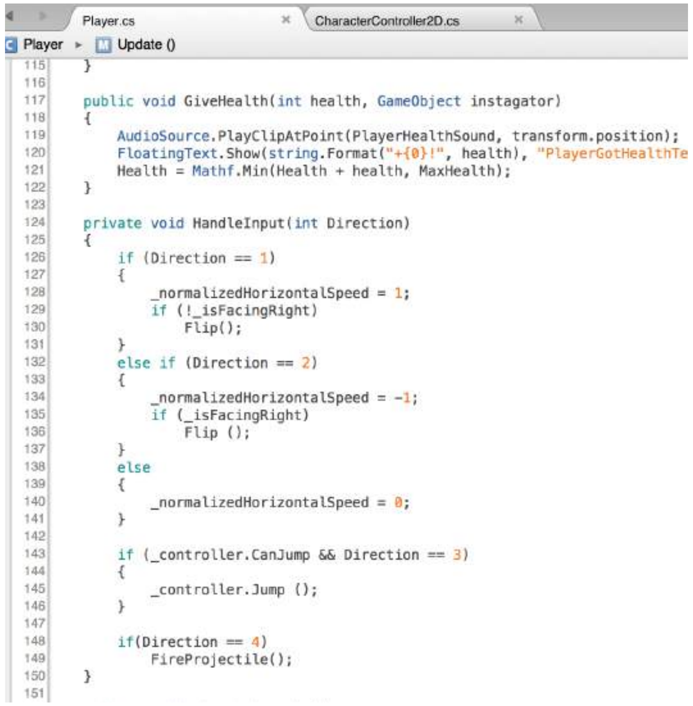
Once I had defined the "1, 2, 3 and 4" inputs in Arduino. Back in Unity I needed the game engine to read from the Arduino, so the USB port it is connected to.

I then changed the input in the "Player.cs" script according to these numbers from the Arduino, so when a connection is made, the player would do an action.

(Please see version 1 of my prototype on page 153 to see how this test went).



```
using UnityEngine;
using System.Collections;
using System.Net;
using System.IO.Ports;

public class Player : MonoBehaviour
{
    SerialPort sp = new SerialPort("/dev/cu.usbmodem1451", 9600);

    private bool _isFacingRight;
    private CharacterController2D _controller;
    private float _normalizedHorizontalSpeed;

    public float MaxSpeed = 8;
    public float SpeedAccelerationOnGround = 10f;
    public float SpeedAccelerationInAir = 5f;
    public int MaxHealth = 100;
    public GameObject OuchEffect;
    public Projectile Projectile;
    public float FireRate;
    public Transform ProjectileFireLocation;
    public GameObject FireProjectileEffect;
    public Animator Animator;
    public AudioClip PlayerHitSound;
    public AudioClip PlayerShootSound;
    public AudioClip PlayerHealthSound;

    public int Health { get; private set; }
    public bool IsDead { get; private set; }

    private float _canFireIn;

    void Start()
    {
        sp.Open();
        sp.ReadTimeout = 1;
    }

    public void Awake()
```



```
    }

    public void GiveHealth(int health, GameObject instagator)
    {
        AudioSource.PlayClipAtPoint(PlayerHealthSound, transform.position);
        FloatingText.Show(string.Format("+{0}!", health), "PlayerGotHealthTe
        Health = Mathf.Min(Health + health, MaxHealth);
    }

    private void HandleInput(int Direction)
    {
        if (Direction == 1)
        {
            _normalizedHorizontalSpeed = 1;
            if (!_isFacingRight)
                Flip();
        }
        else if (Direction == 2)
        {
            _normalizedHorizontalSpeed = -1;
            if (_isFacingRight)
                Flip ();
        }
        else
        {
            _normalizedHorizontalSpeed = 0;
        }

        if (_controller.CanJump && Direction == 3)
        {
            _controller.Jump ();
        }

        if(Direction == 4)
            FireProjectile();
    }

```

GLOVE CONTROLLER

Glove works as it should and has been prototyped (page 153).

Now I want to implement an accelerometer as I feel that from the prototyping on version 1 it would make it easier to move the player backwards and forwards, and more exercising of the hand muscles is being used.

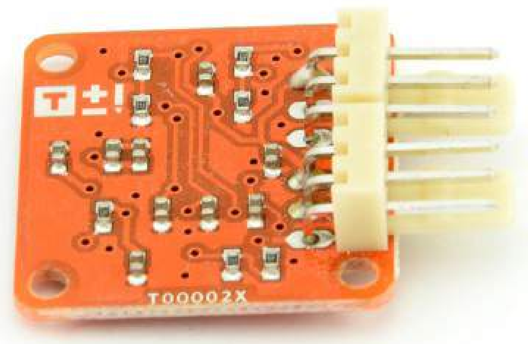
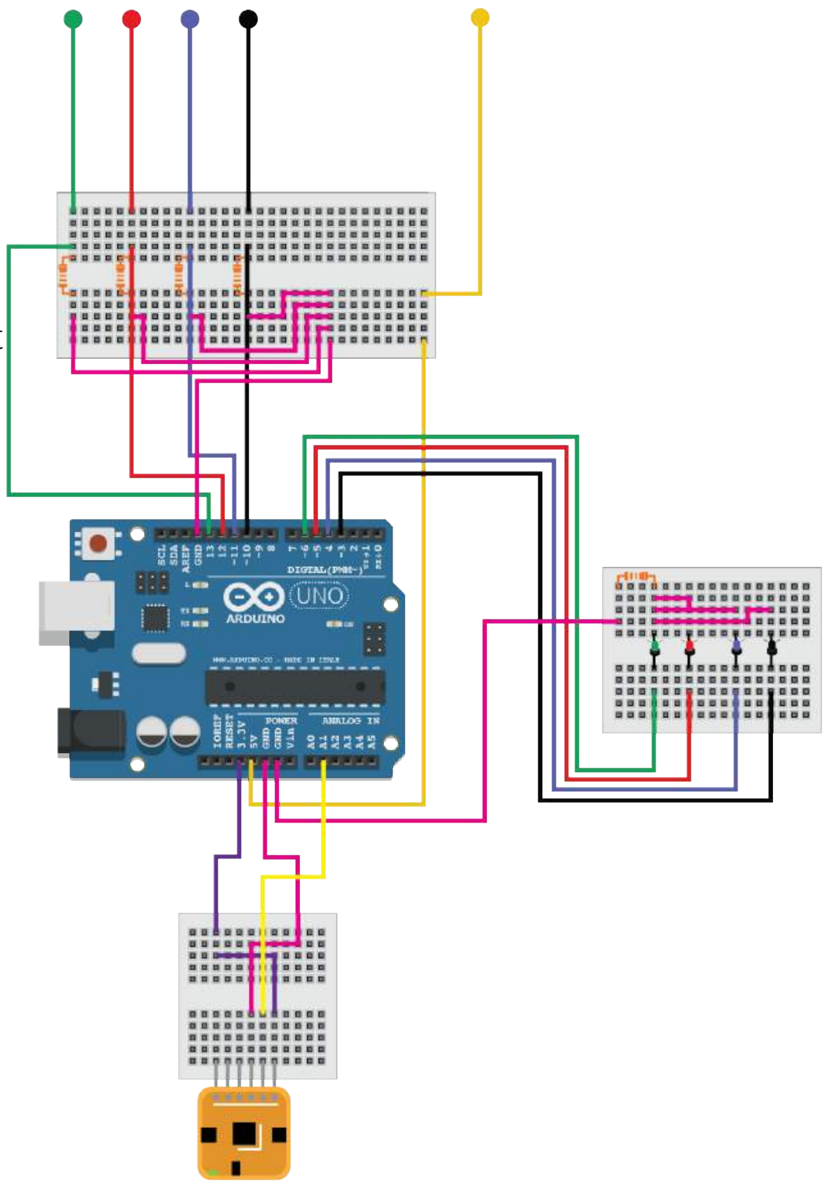


Image used: http://www.kiwi-electronics.nl/image/cache/data/products/arduino/tinkerkit/T000020-2-800x533.jpg

GLOVE CONTROLLER

I had a diagram mocked up of how I think the accelerometer would work in my glove.

Connecting only the Y-axis as this is the only one i want to measure, I connect it to the existing Arduino circuit and it will hopefully work together.



KEY:

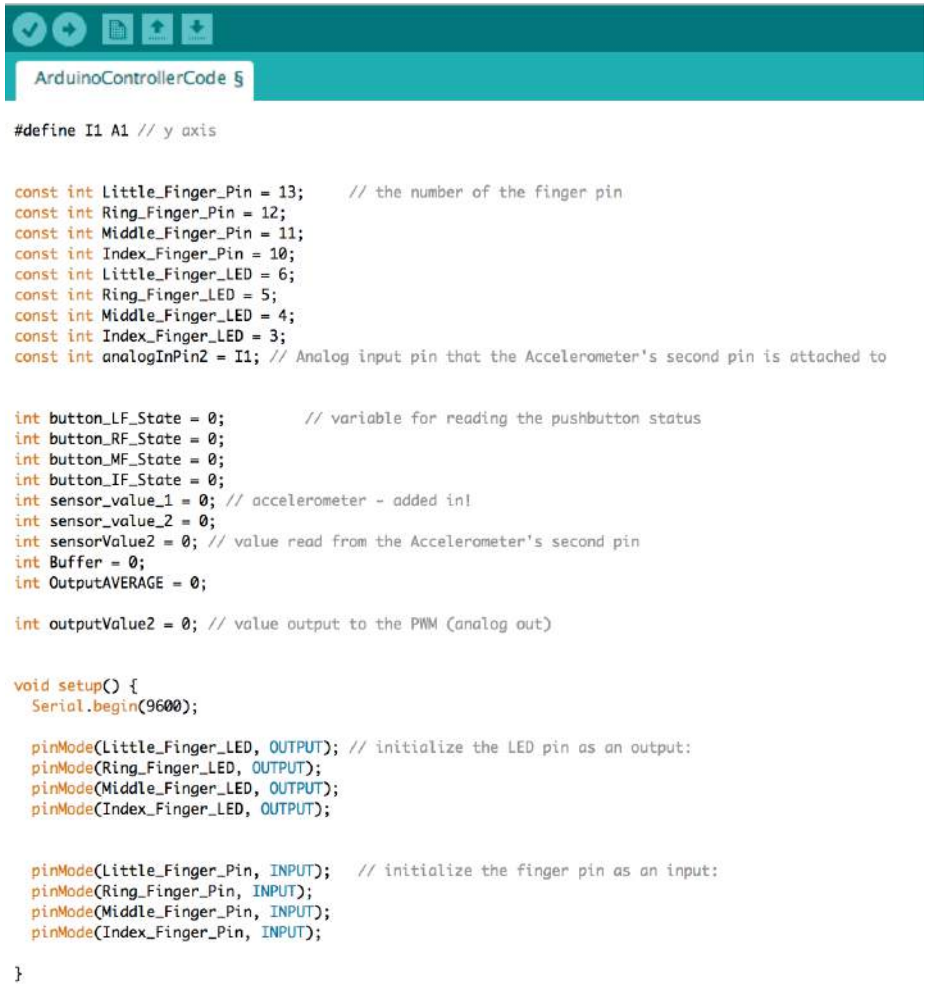
🟩	LITTLE FINGER
🟥	RING FINGER
🟦	MIDDLE FINGER
⬛	INDEX FINGER
🟨	THUMB (5V)
🟥	GND
🟪	ACCELEROMETER (3.3V)
🟨	ACCELEROMETER (Y AXIS)

GLOVE CONTROLLER

(Part 1)

From my existing code I already have, I added in extra code that would control the accelerometer the way I want it to.

There are similarities to the added code to the pervious code I have implemented so this was not too complicated to understand.



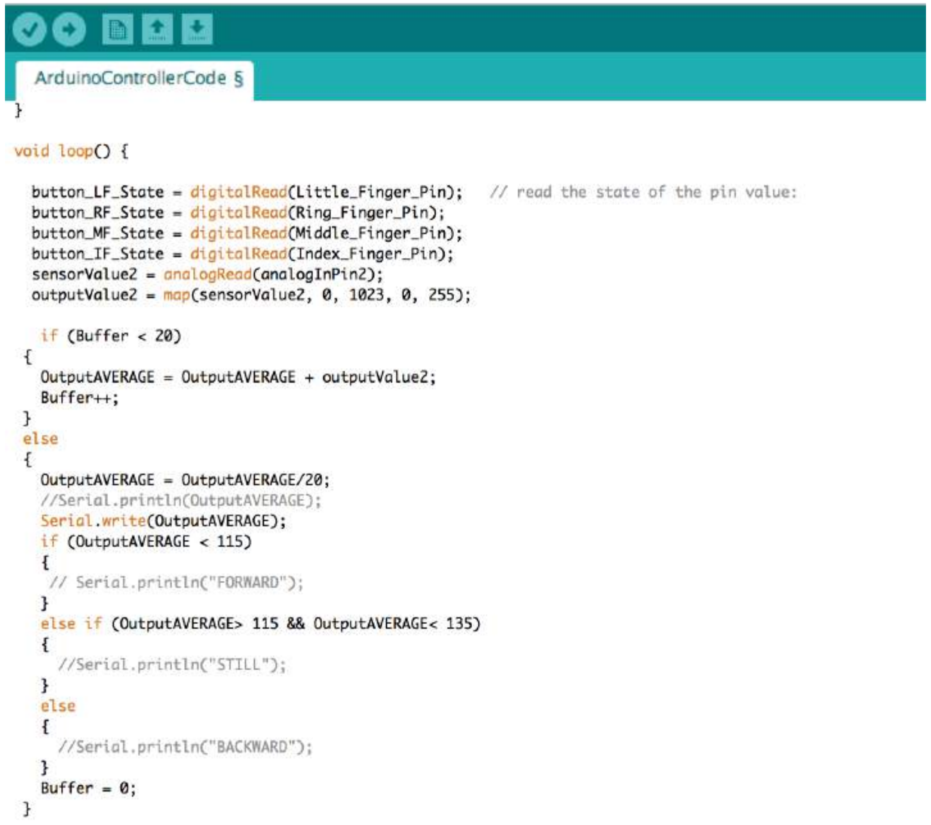
```
#define I1 A1 // y axis


const int Little_Finger_Pin = 13;     // the number of the finger pin
const int Ring_Finger_Pin = 12;
const int Middle_Finger_Pin = 11;
const int Index_Finger_Pin = 10;
const int Little_Finger_LED = 6;
const int Ring_Finger_LED = 5;
const int Middle_Finger_LED = 4;
const int Index_Finger_LED = 3;
const int analogInPin2 = I1; // Analog input pin that the Accelerometer's second pin is attached to


int button_LF_State = 0;         // variable for reading the pushbutton status
int button_RF_State = 0;
int button_MF_State = 0;
int button_IF_State = 0;
int sensor_value_1 = 0; // accelerometer - added in!
int sensor_value_2 = 0;
int sensorValue2 = 0; // value read from the Accelerometer's second pin
int Buffer = 0;
int OutputAVERAGE = 0;

int outputValue2 = 0; // value output to the PWM (analog out)


void setup() {
  Serial.begin(9600);

  pinMode(Little_Finger_LED, OUTPUT); // initialize the LED pin as an output:
  pinMode(Ring_Finger_LED, OUTPUT);
  pinMode(Middle_Finger_LED, OUTPUT);
  pinMode(Index_Finger_LED, OUTPUT);


  pinMode(Little_Finger_Pin, INPUT);   // initialize the finger pin as an input:
  pinMode(Ring_Finger_Pin, INPUT);
  pinMode(Middle_Finger_Pin, INPUT);
  pinMode(Index_Finger_Pin, INPUT);

}
```

(Part 2)

I need to find the average number that the Y-axis emits because the values is currently gives out are not clear at all.

Once the average umber has been found an ‘if statement’ is needed.

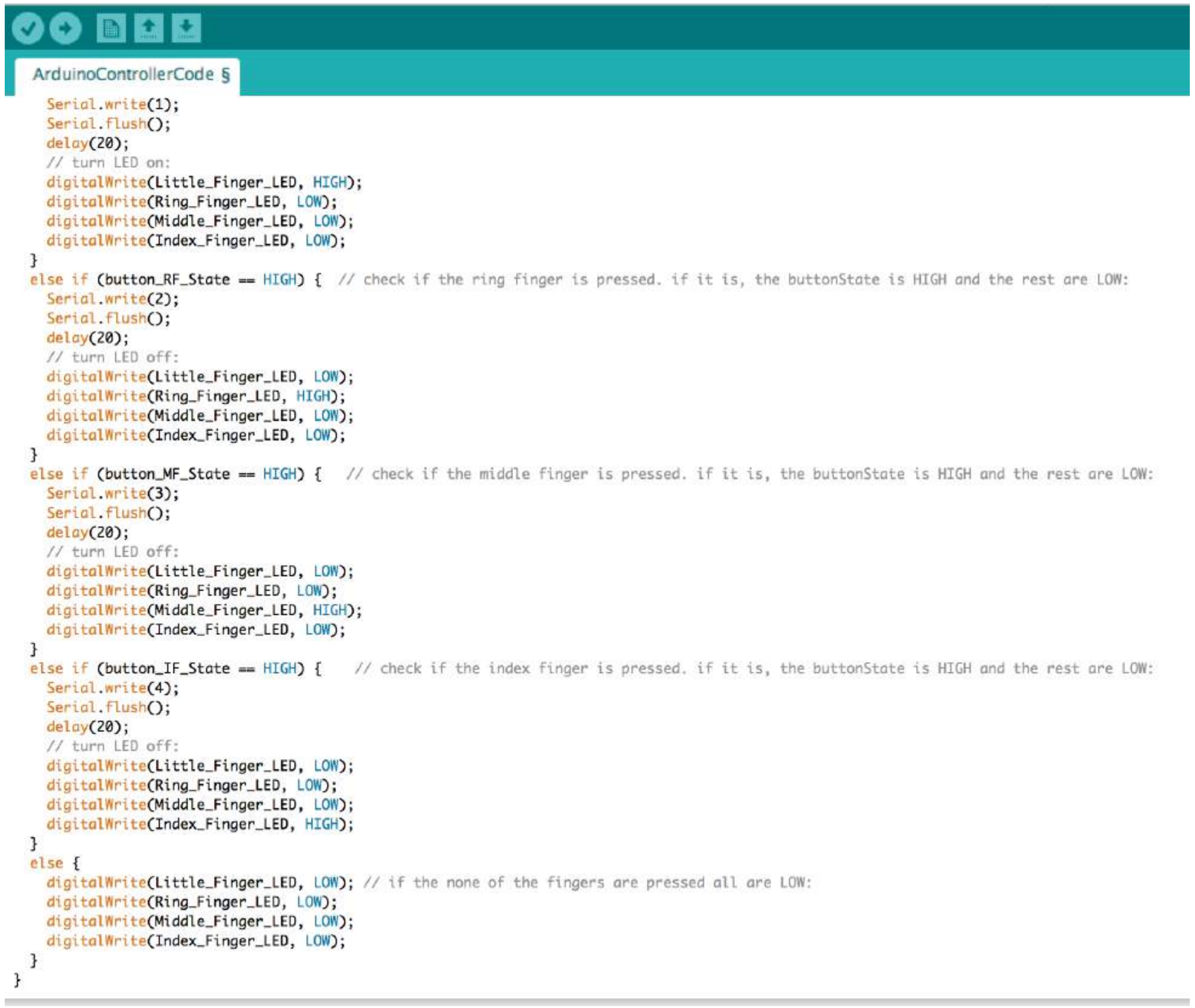
If the numbers given out are between 2 values it will print “Forward” (or less than 115 for example). This is what will be used in Unity to determine the controls.



```
}

void loop() {

  button_LF_State = digitalRead(Little_Finger_Pin);   // read the state of the pin value:
  button_RF_State = digitalRead(Ring_Finger_Pin);
  button_MF_State = digitalRead(Middle_Finger_Pin);
  button_IF_State = digitalRead(Index_Finger_Pin);
  sensorValue2 = analogRead(analogInPin2);
  outputValue2 = map(sensorValue2, 0, 1023, 0, 255);

   if (Buffer < 20)
 {
   OutputAVERAGE = OutputAVERAGE + outputValue2;
   Buffer++;
 }
 else
 {
   OutputAVERAGE = OutputAVERAGE/20;
   //Serial.println(OutputAVERAGE);
   Serial.write(OutputAVERAGE);
   if (OutputAVERAGE < 115)
   {
    // Serial.println("FORWARD");
   }
   else if (OutputAVERAGE> 115 && OutputAVERAGE< 135)
   {
     //Serial.println("STILL");
   }
   else
   {
     //Serial.println("BACKWARD");
   }
   Buffer = 0;
 }
```

(Part 3)

The code here is unchanged, however this is the end of the full code for my Arduino glove controller.



```
    Serial.write(1);
    Serial.flush();
    delay(20);
    // turn LED on:
    digitalWrite(Little_Finger_LED, HIGH);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_RF_State == HIGH) {  // check if the ring finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.write(2);
    Serial.flush();
    delay(20);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, HIGH);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_MF_State == HIGH) {   // check if the middle finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.write(3);
    Serial.flush();
    delay(20);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, HIGH);
    digitalWrite(Index_Finger_LED, LOW);
  }
  else if (button_IF_State == HIGH) {    // check if the index finger is pressed. if it is, the buttonState is HIGH and the rest are LOW:
    Serial.write(4);
    Serial.flush();
    delay(20);
    // turn LED off:
    digitalWrite(Little_Finger_LED, LOW);
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, HIGH);
  }
  else {
    digitalWrite(Little_Finger_LED, LOW); // if the none of the fingers are pressed all are LOW:
    digitalWrite(Ring_Finger_LED, LOW);
    digitalWrite(Middle_Finger_LED, LOW);
    digitalWrite(Index_Finger_LED, LOW);
  }
}
```

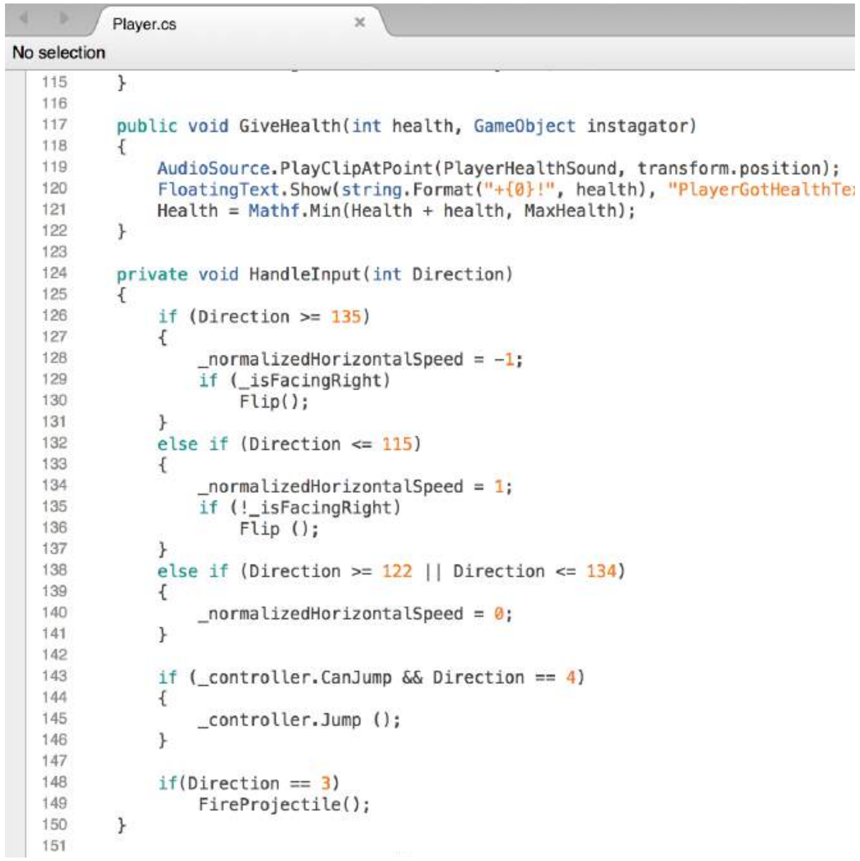
GLOVE CONTROLLER

Once I had the values figured out from the Arduino, it was time to place the movement inputs in the "Player.cs" script.

These are just if statements similar to the Arduino code that read if they are more than 135, go backwards.

If the value is less than 115, go forwards. However if the values are between 122 and 134 then stand still.

A test on version 3 of the prototype can be found on 155.



```
}

public void GiveHealth(int health, GameObject instagator)
{
    AudioSource.PlayClipAtPoint(PlayerHealthSound, transform.position);
    FloatingText.Show(string.Format("+{0}!", health), "PlayerGotHealthTe
    Health = Mathf.Min(Health + health, MaxHealth);
}

private void HandleInput(int Direction)
{
    if (Direction >= 135)
    {
        _normalizedHorizontalSpeed = -1;
        if (_isFacingRight)
            Flip();
    }
    else if (Direction <= 115)
    {
        _normalizedHorizontalSpeed = 1;
        if (!_isFacingRight)
            Flip ();
    }
    else if (Direction >= 122 || Direction <= 134)
    {
        _normalizedHorizontalSpeed = 0;
    }

    if (_controller.CanJump && Direction == 4)
    {
        _controller.Jump ();
    }

    if(Direction == 3)
        FireProjectile();
}

```

GLOVE CONTROLLER

- CONCLUSION OF SECTION

The building of my glove was quite a challenge in parts, such as getting the serial port to be read in Unity but in the end, i managed to get it done.

I feel like adding the accelerometer in as a little extra has really improved the overall value of my glove because there are now more exercises to perform while wearing the glove and the more ways to interact with it, the better it becomes.

PROTOTYPING

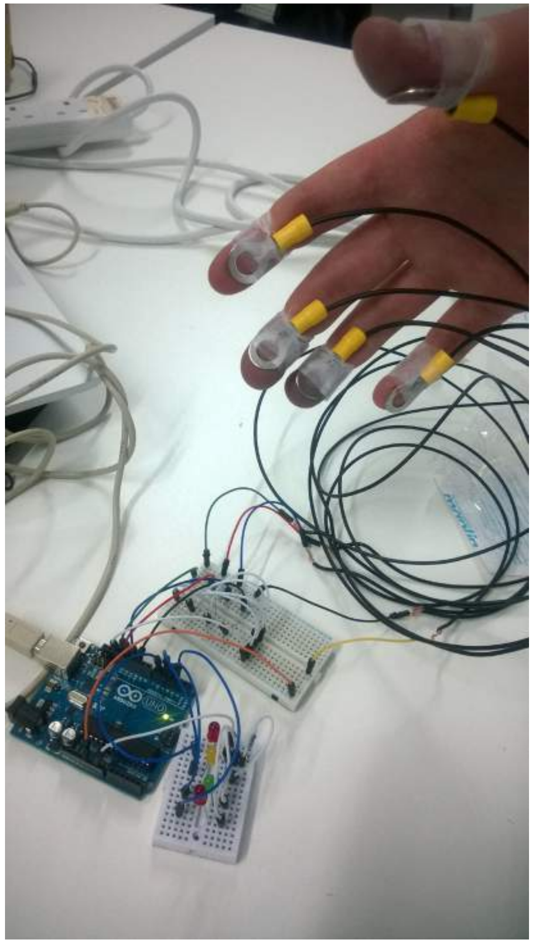
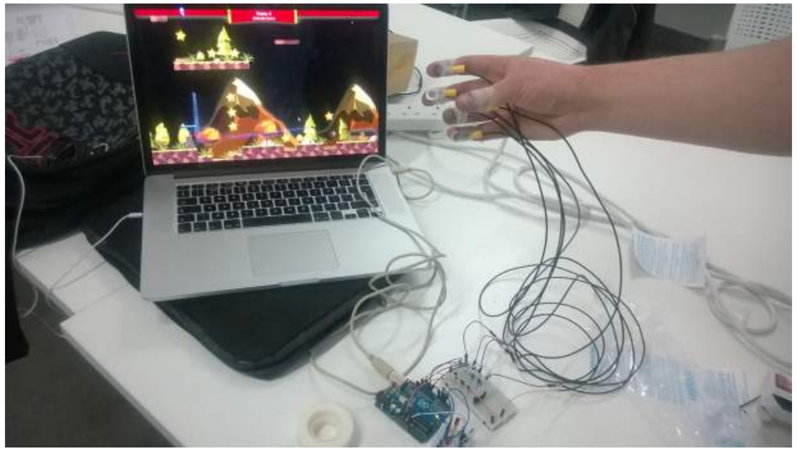
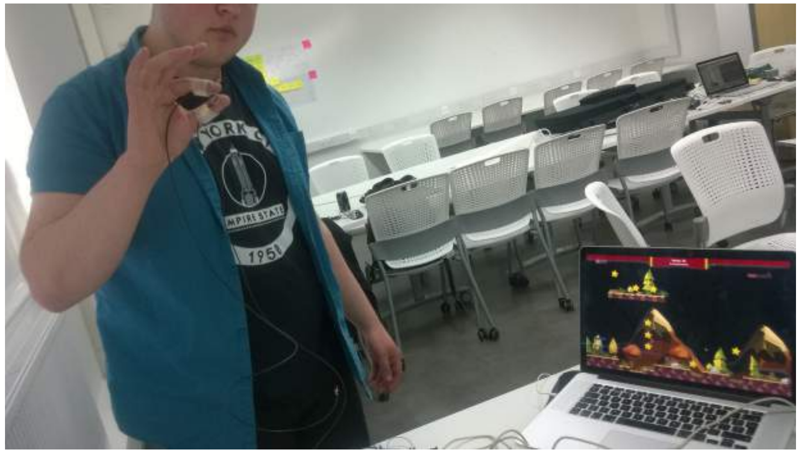
Will it all work together?

PROTOTYPING

VERSION 1

For my very first prototype, I built the controller outside of the glove just to see if the connections actually work and that my game is playable. I simply attached my touch points to my fingers and launched the game.

It was quite overwhelming seeing my idea brought to life as well as I had seen. Yes, there may be a few issues with the level being a little complex (I can soon change this) but the main controller is working, as I want it to.

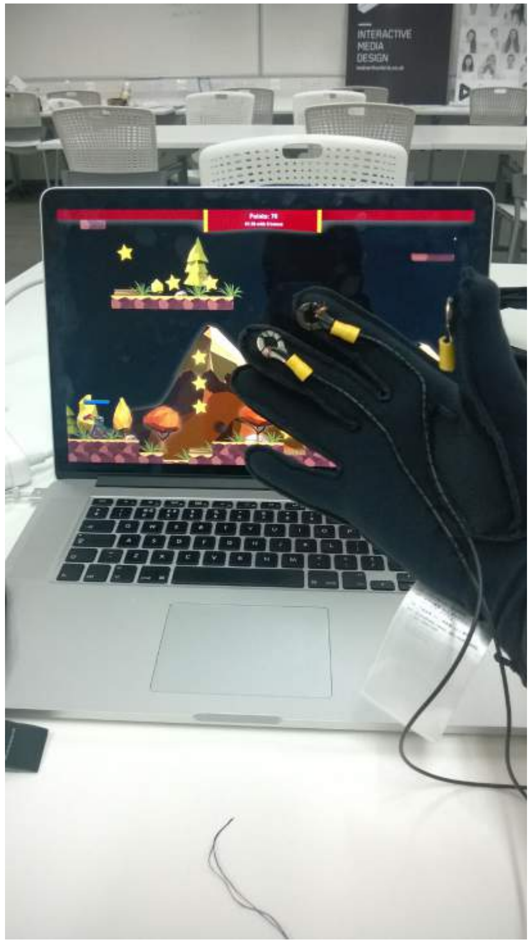




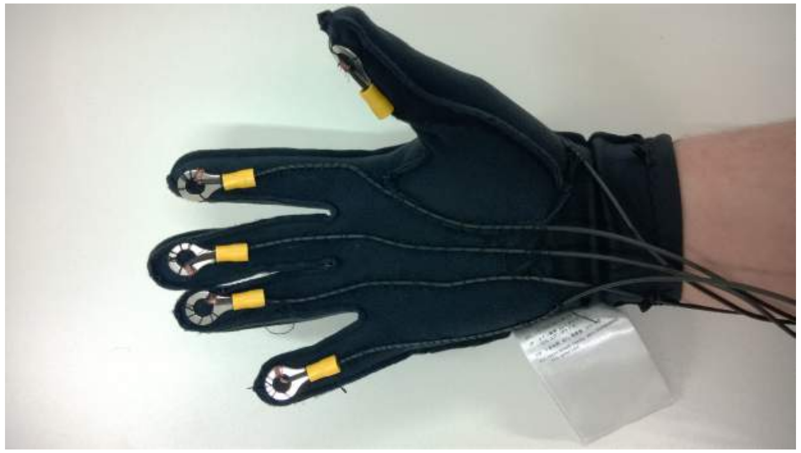
VERSION 2

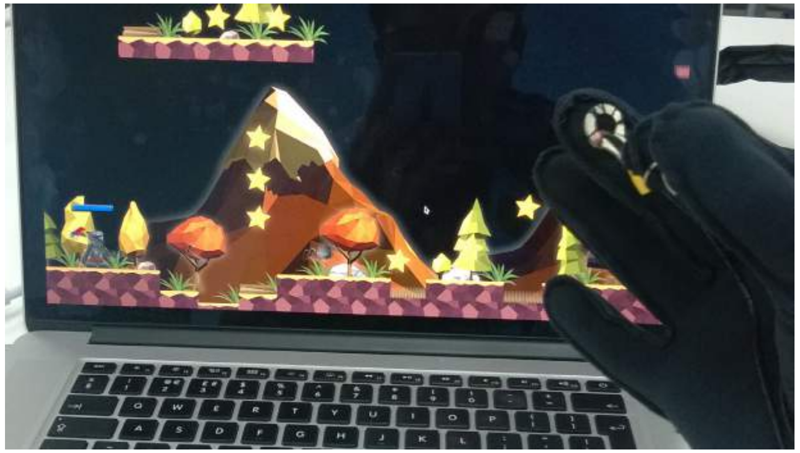
I was really pleased with how my first prototype worked, better than I had expected really.

For my next prototype I decided to stich it to the glove, Turning the left glove inside out and sticking around the trim line of the glove to make the finger movement easy to perform, this worked very effective.

I have plans that when I am happy with the final wiring I will put the right hand glove over the top to hide all of the 'ugly' wires so it feels like a more professional product overall. I will be doing this in the next couple of prototypes.





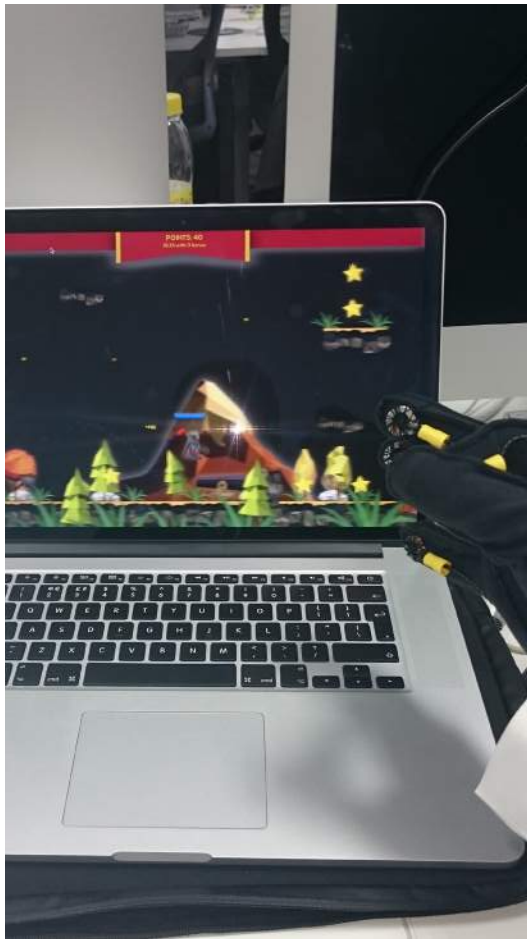
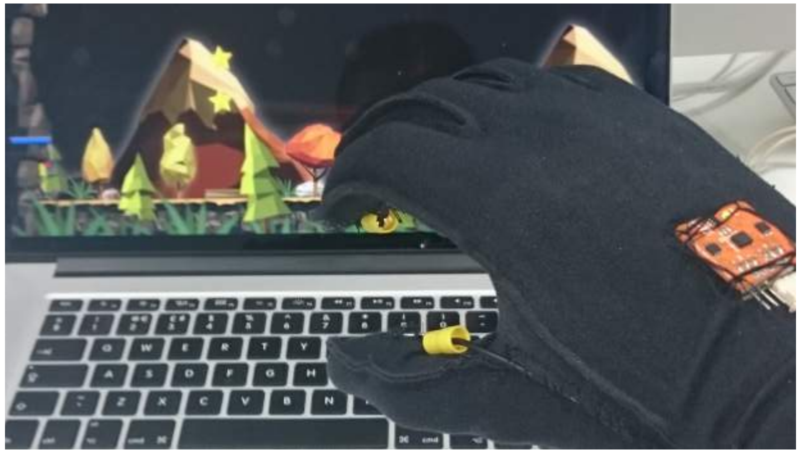
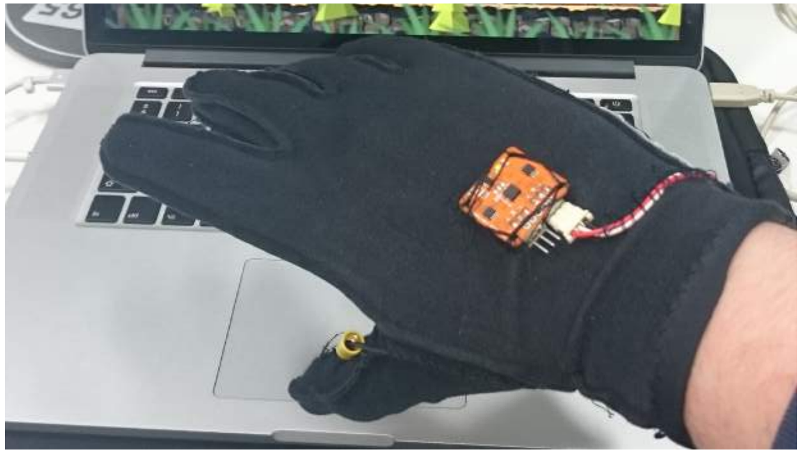


PROTOTYPING

VERSION 3

Testing the game and controller together with the accelerometer was a lot of fun. Along with a little refresh of my graphics for the game this is where I first saw a glimpse of a product that was almost there.

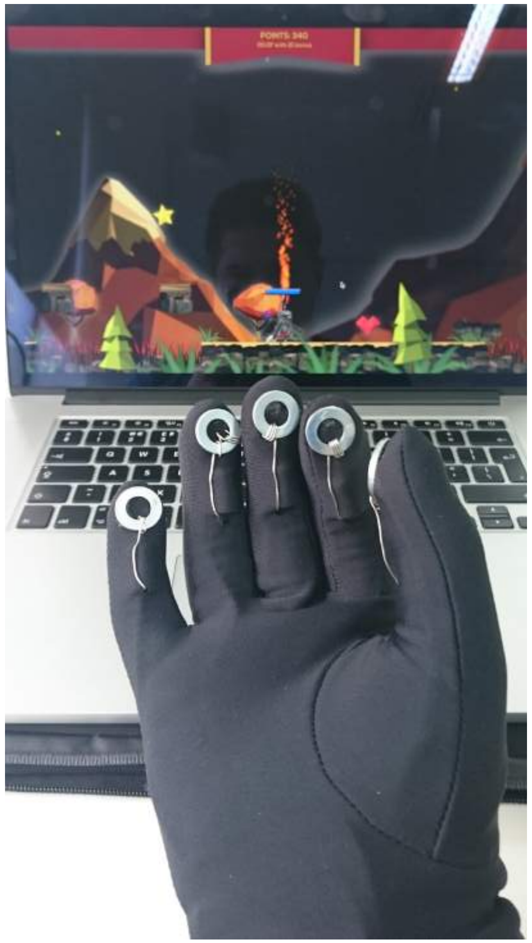
I am happy with the way the controller works and the build is almost there,

VERSION 4

From version 3 I knew that everything worked, as it should.

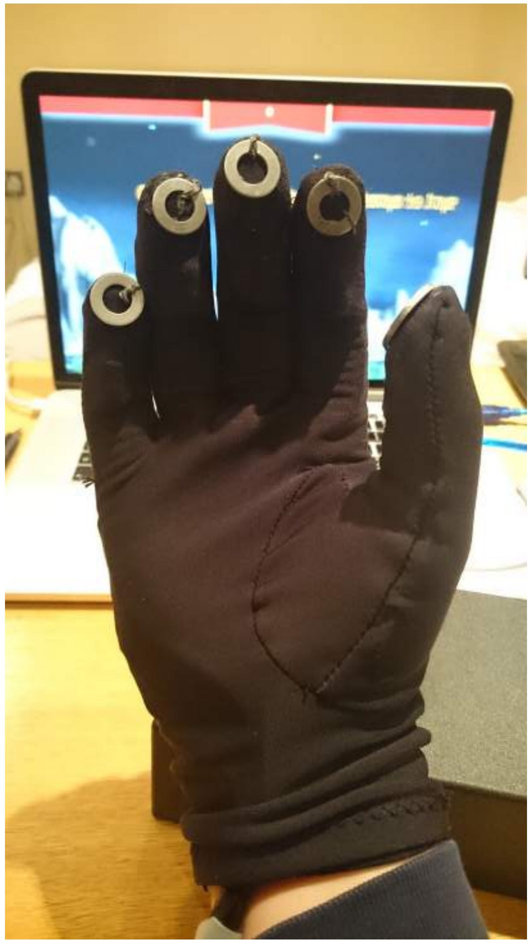
Version 4 was a more aesthetic clean up of the glove. I have covered up the wiring with the second glove over the top and 5 metal rings are visible which looks easier to understand than my last prototype.



VERSION 5

In version 4 of my prototype I thought I had cracked my glove design but after many attempts to play, the wires snapped! This was a bit of a pain but I used conductive thread instead which seemed to fix my issue.

Also you may notice I also changed a little bit of the design of the game to make it more of a 'night time' setting with smoke and darker colours, which from feedback looked so much better.



PROTOTYPING

- CONCLUSION OF SECTION

Through out my prototyping process I was still building my game and glove, and the 4 iterations of prototyping I did hopefully show this.

I am confident that my version 4 prototype is fit for a user test as a final product.

The feedback that I have been given to improve on my design of not only the glove but the game has been very beneficial as I feel it looks and plays a lot more simpler than before.

Prototyping has played a big part in getting my design right for the user as it needs to be easy and comfortable and now, i feel like it is and by having all of the wires hidden away in the glove, it looks more professional.

TESTING

What do my audience think?

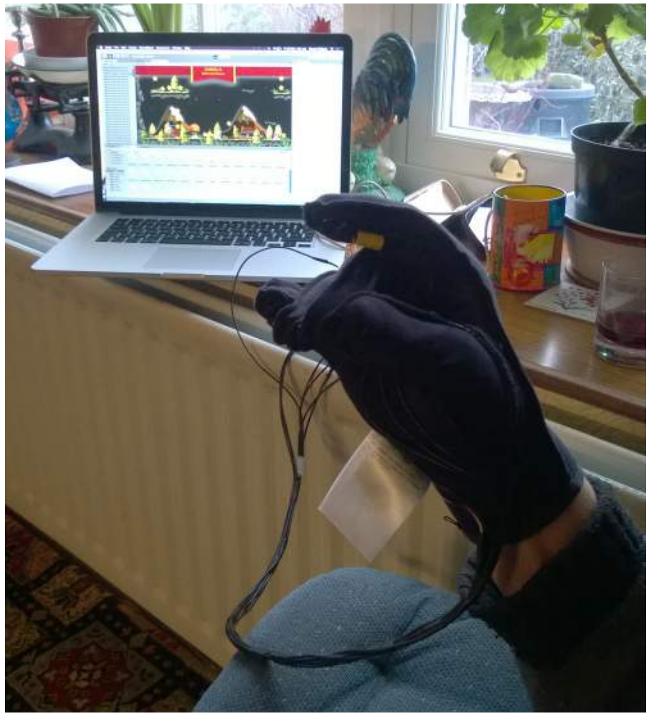
TESTING

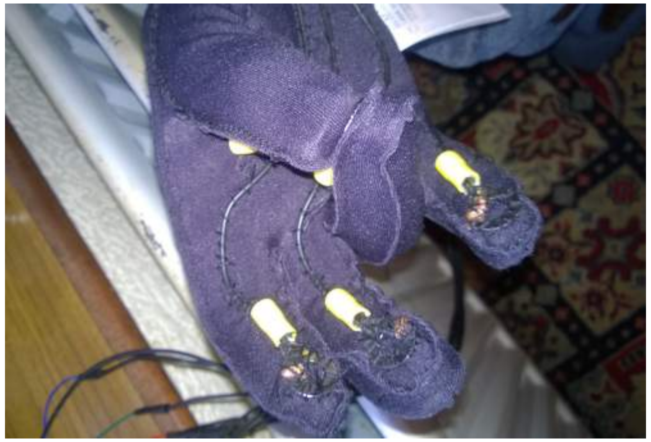
- FIRST TEST (USING VERSION 3 PROTOTYPE)

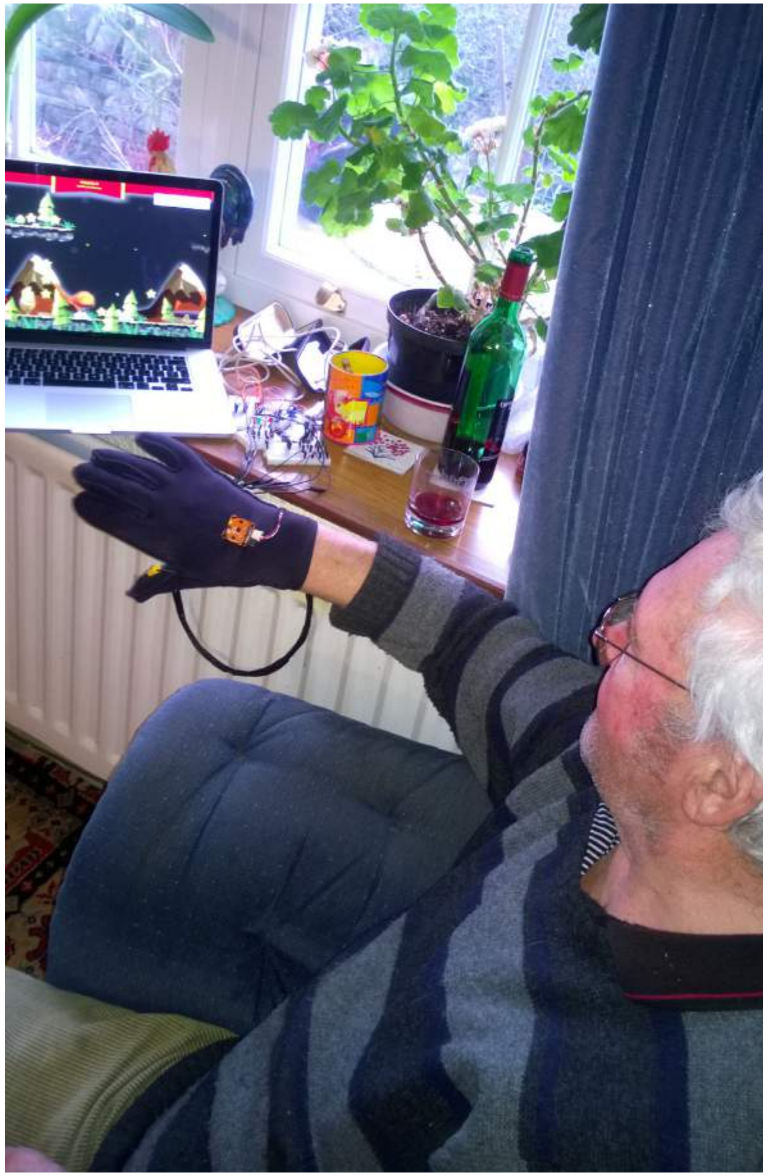
My first test was a success I have to admit; the controller worked really well when my Grandad had a play with it.

The finger movements were very similar to that of the Arthritis exercises he does and he was able to move the player across the world.

Only a couple of issues were flagged up, the thumb connector was a little low so was sometimes hard to reach and maybe I could make the level a little more simpler to move across as I am making this for the older generation primarily.







TESTING

- SECOND TEST (USING VERSION 5 PROTOTYPE)

When I let my Grandad have another go with my improved glove controller and game the feedback was great.

I was worried that by having 2 gloves stitched together would be a bit difficult getting it on, it turned out to be fine and "it fit like a glove" it was very comfy and the controls were a lot easier to use in this improved version.

TESTING

- CONCLUSION OF SECTION

After my testing stages I have decided that my glove is an effective prototype for people who are affected by Arthritis.

The glove, which has seen many changes and improvements now works as I imagined it at the start of the development stage because it does exercise your finger movements and it does help people with Arthritis keep there joints moving because if the exercises are not performed, their joints seize up and become incredibly numb.

I have seen for my own eyes that by testing, this glove works.

BRANDING

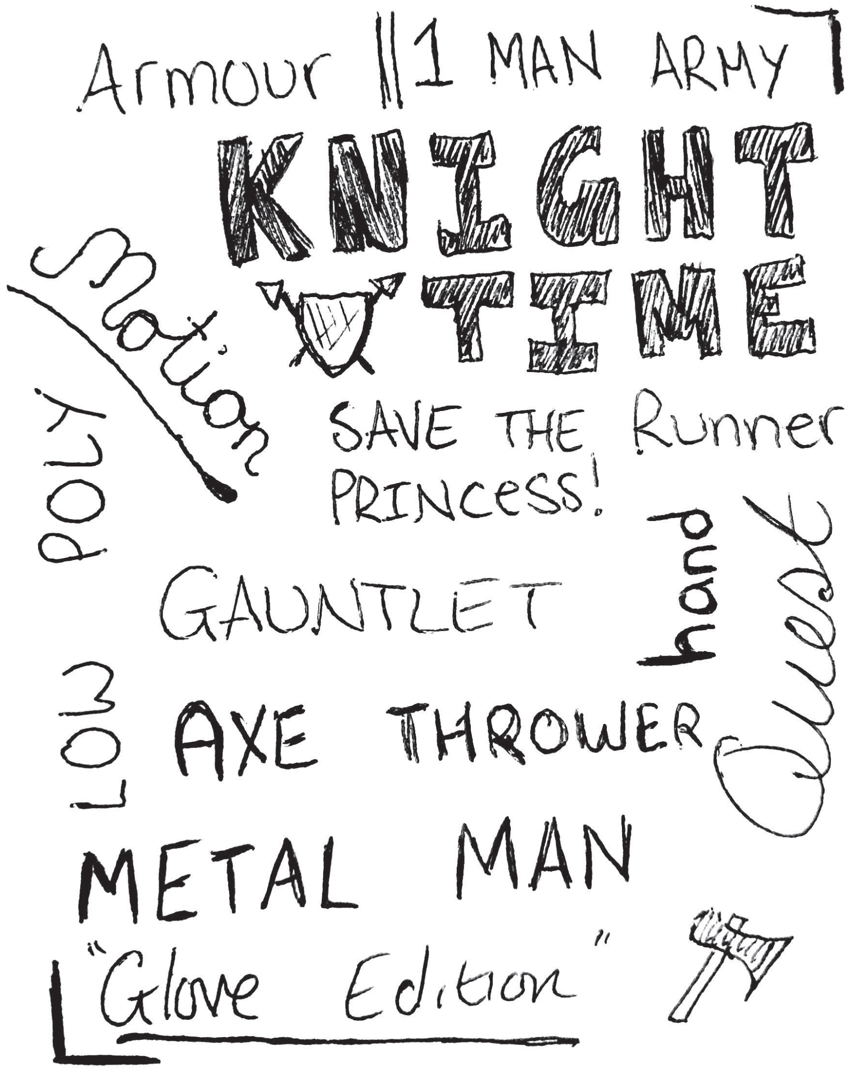
Lets give it a name!

BRANDING

NAME GENERATION

I need a name for my game and also the glove controller as a whole package.

From a little exploration of words that I feel my game represents I narrowed down "Knight Time" as my name of choice as I felt this tied up with the feel and style. You play as a knight and also since late on in the development stage I changed the setting to 'night time' so I felt this was the best way forward.



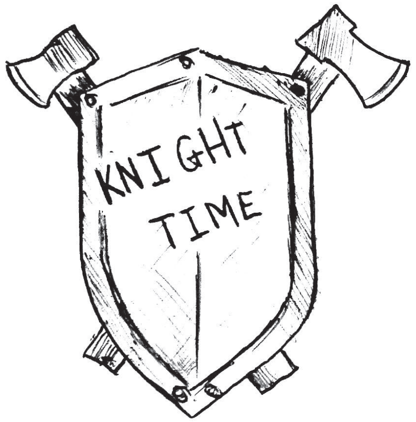
BRANDING

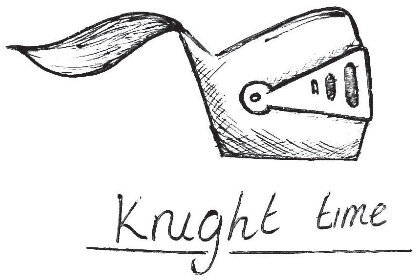
LOGO DEVELOPMENT

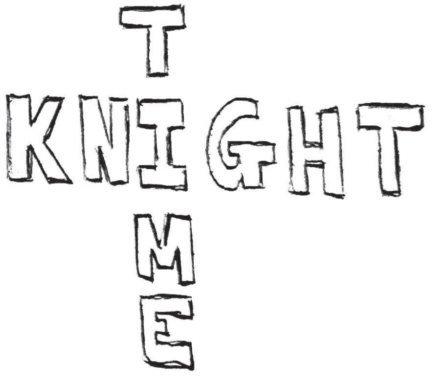
I have the name, now I need to think about a logo that will be at the forefront of my overall package.

I had 3 styles, which I thought worked well, a shield and axe, the knight's head and a typography logo.

With a bit of digital tweaking I think they could all work well, however I am striving towards my first design, as I prefer this one as it makes a bolder statement.



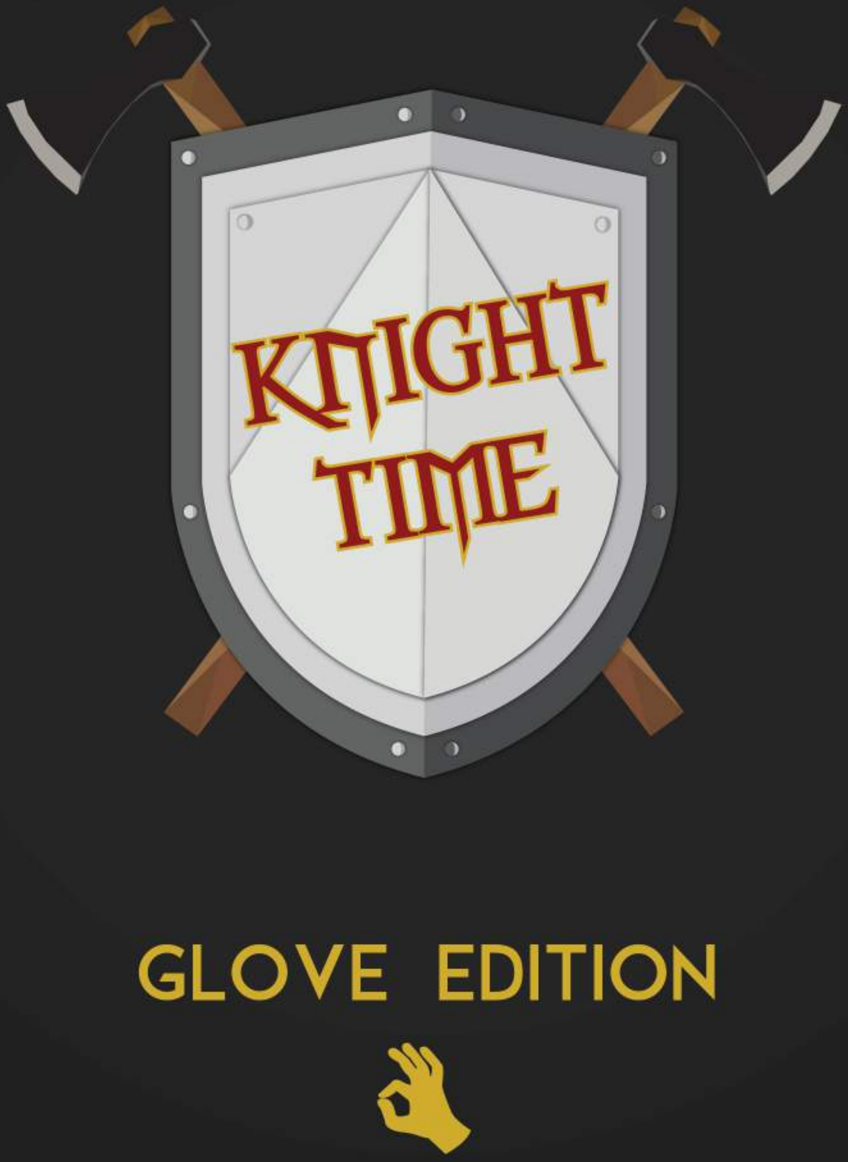




LOGO DEVELOPMENT

Modifying my sketch digitally in Illustrator I have created a logo in which I am very happy with.

Also with the added "Glove Edition" this gives the indication that this is how you play the game which is a little detail I have thought about adding for a while and I think it works.

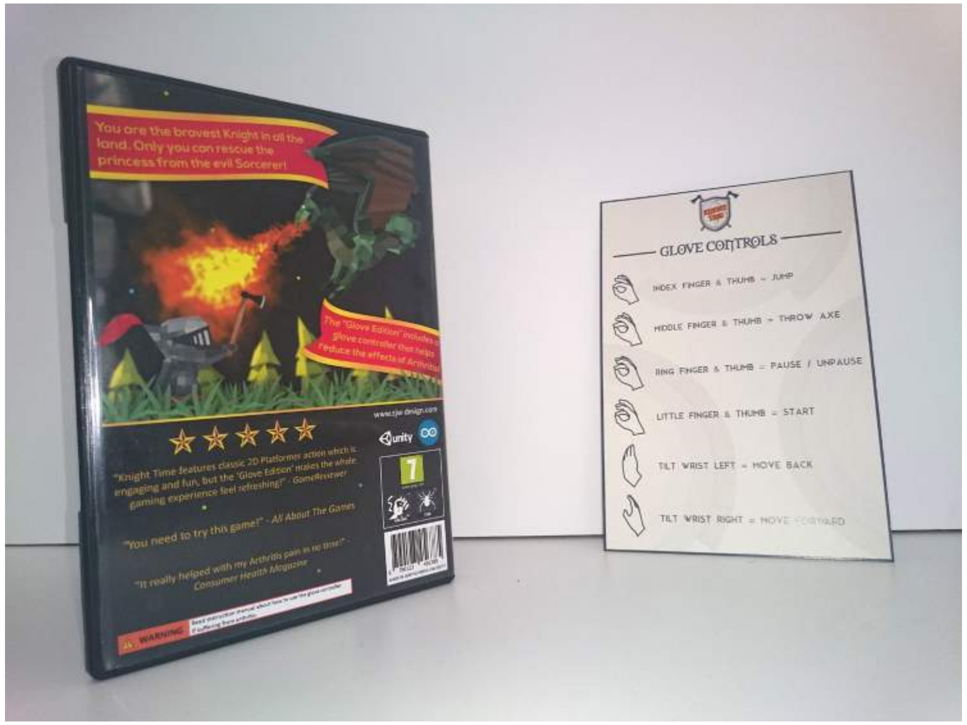


BRANDING

PACKAGING

To package the game up to a standard that i was happy with I wanted it in a DVD case, I designed a sleeve to go in it and made it look as authentic as I could to that of a real published game.

I thought it would be handy to include a little leaflet with the controls on too, so this fits inside the case and it very easy to follow. I feel like the designs I have created reflect the game as a whole and it all ties my project up together nicely.


You are the bravest Knight in all the land. Only you can rescue the princess from the evil Sorcerer!
The "Glove Edition" includes a glove controller that helps reduce the effects of Arthritis
"You need to try this game!" - All About The Games
GLOVE CONTROLS
INDEX FINGER & THUMB = JUMP
MIDDLE FINGER & THUMB = THROW AXE
RING FINGER & THUMB = PAUSE / UNPAUSE
LITTLE FINGER & THUMB = START
TILT WRIST LEFT = MOVE BACK
TILT WRIST RIGHT = MOVE FORWARD

BRANDING

- CONCLUSION OF SECTION

Seeing my project come together in one nice neat package does put a smile on my face. From my very first idea to seeing it all there and ready to use does wrap my project up.

I feel like the logo and overall colour scheme matches that of my game so nothing feels alienated. I tried my hardest to make everything seem like a legitimate product and I think, as well as others, looks professional.

CONCEPT VIDEO

Promoting my innovation

CONCEPT VIDEO

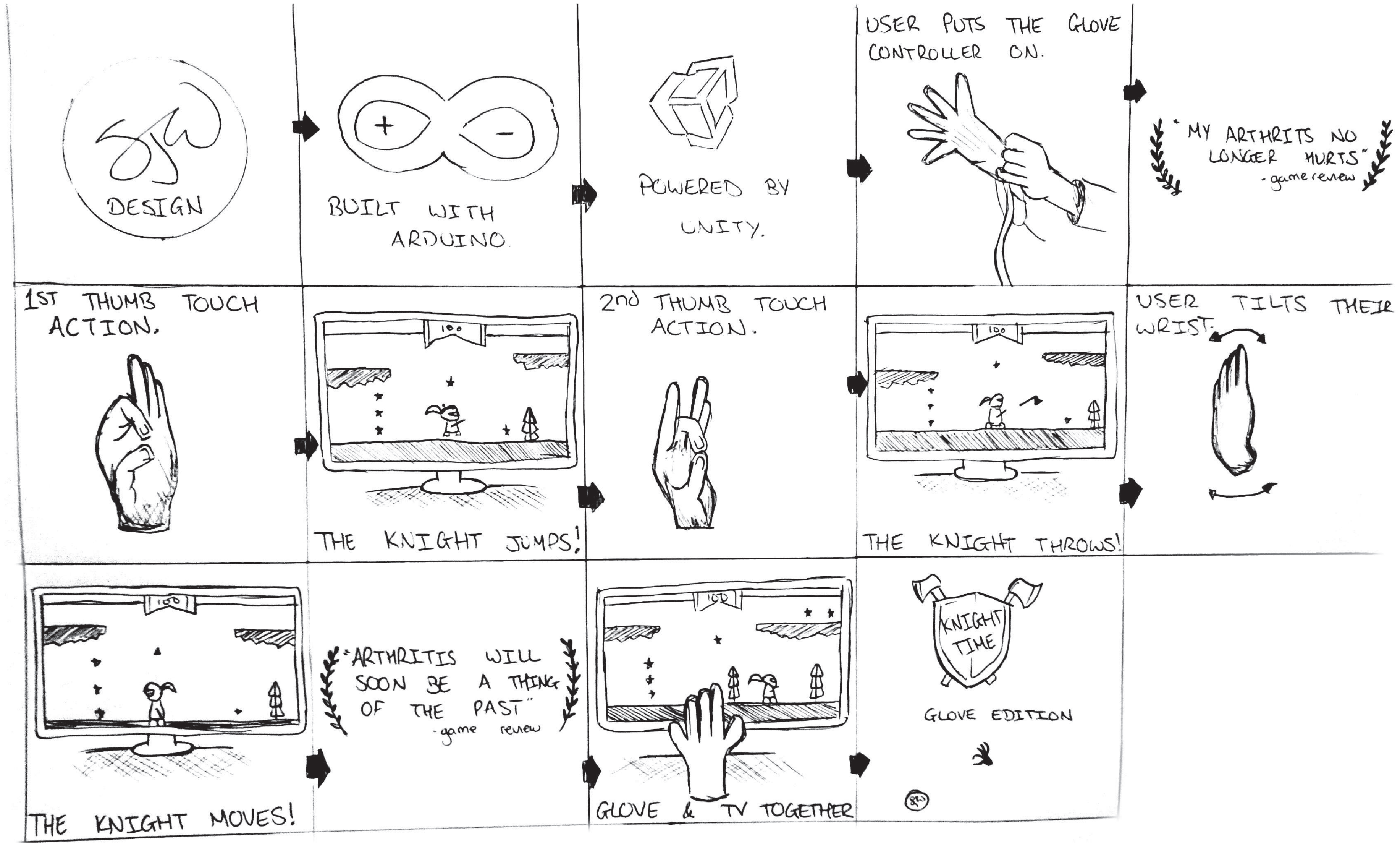
- PLANNING

For a concept video I want it to show the best parts of my project, so how the game element is a fun way to exercise your fingers and how effective it can be, especially for the older generation.

I want to try and make my video like that of an official video game trailer. So have opening credits, review quotes, gameplay and seeing the controller in action. To do this, the best way is to make a story board and make sure i include all of the best parts of my design.

CONCEPT VIDEO

- STORY BOARD



CONCEPT VIDEO

- FINAL VIDEO

Filming for the final video was really fun. I strapped a GoPro Hero 3 to myself and played Knight Time on the TV. This footage is to represent how the game is played from the users perspective.

In After Effects I compile the video together to create a video game trailer like result that I am very satisfied with. I think it demonstrates my project well and follows industry standards of how to show off a game that is yet to be released.

The video can be viewed here:

https://www.youtube.com/watch?v=F2vuFiGf--0

FINAL COMMENTS

Wrapping things up...

FINAL COMMENTS

From something I built ground up, I am extremely proud of myself. Making a game is something in which I have always wanted to acheive and i think that learning C# has been very beneficial to my skill set.

This document has shown the lengthy process I've taken to get to my final product, a lot of time and effort has been spent and i feel as if I've shown all of my hard work for my final project. A lot of coding has been done which is something I wanted to focus on, as I wanted to get better at it and not just design an interface.

From my vision I outlined at the beginning of the project, it had to be iterated a lot in order for me to be happy with the final result. I believe that my solution to the design problem of 'when people suffer from Arthritis there are not many motivations to help them get better' has been tackled as my glove controller is that motivation that has been missing. From continuous feedback throughout I have designed a solution around that problem so the people who need it most get the most benefit from it.

Of course, there is always room for improvement and I have many ideas for future versions of my glove; I would perhaps exchange the metal ring touch points for a conductive fabric to make it look more like and ordinary glove and have the possibility to make it wireless because nowadays many things are wireless. If I was to change my game, I could do any game I wanted now that I have everything working, maybe I could even create that 3D car game idea I itched on with from the beginning and see how that worked.

But, overall I am really happy with the way I progressed through this module, as it was my last project I wanted to do something beyond my previous skills set to help me get better and I believe that I have gained a lot more skills that will benefit me in the future because the more skills I have under my belt the better, Unity, C# and Arduino have been a difficult learning curve at times but, in the end, beneficial to my independent learning within this final project.