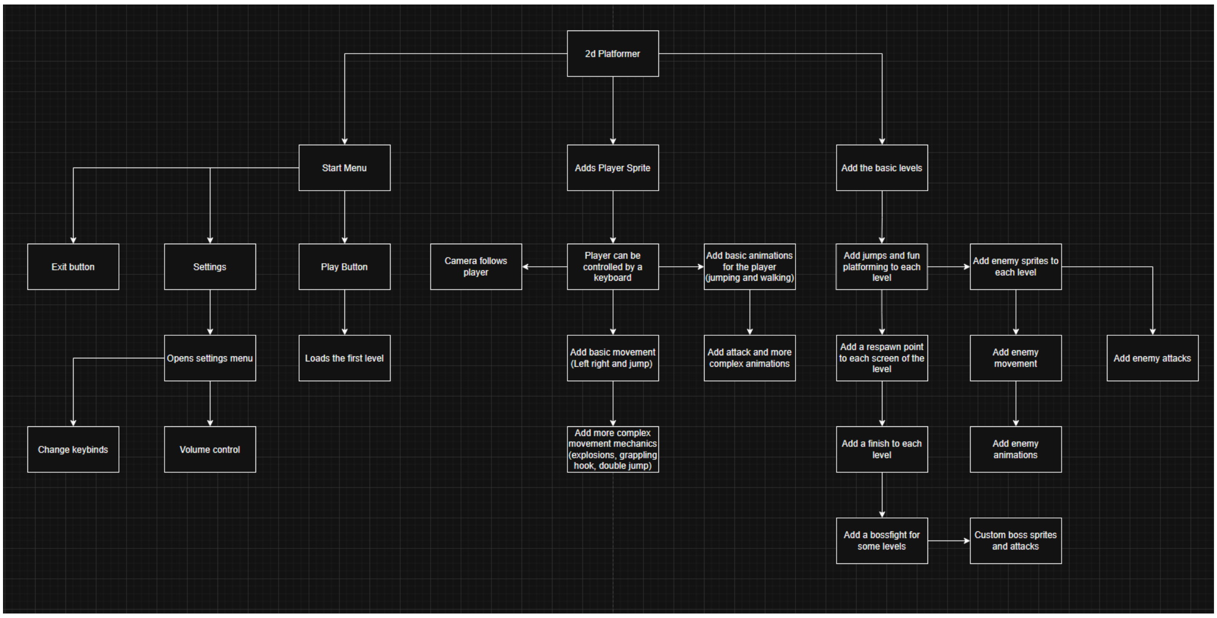
2d Platformer Design:

Structure Diagram:


2d Platformer
Start Menu
Adds Player Sprite
Add the basic levels
Exit button
Settings
Play Button
Camera follows player
Player can be controlled by a keyboard
Add basic animations for the player (jumping and walking)
Add jumps and fun platforming to each level
Add enemy sprites to each level
Opens settings menu
Loads the first level
Add basic movement (Left right and jump)
Add attack and more complex animations
Add a respawn point to each screen of the level
Add enemy movement
Add enemy attacks
Change keybinds
Volume control
Add more complex movement mechanics (explosions, grappling hook, double jump)
Add a finish to each level
Add enemy animations
Add a bossfight for some levels
Custom boss sprites and attacks

This structure diagram is split into 3 parts; these show the overarching components that I want to add to my game. The first part is the start menu, I chose this as it's a fundamental part to the beginning of a game, as it allows the player to change options like volume and controls before the game has started.

TO DO:

ADD IMAGE + EXPLANATION OF STRUCTURE DIAGRAM FOR EACH STAGE OF DEVELOPMENT

TAKE VIDEOS OF PROOF OF MY COMPLETED STAGES

**Stages of Development Plan:**
**Stage 1:**

Before I go in depth for any feature of the project, like mechanics, level design and user interface, I should make a simple version of the game. This will include simple movement and a few jumps to test that the base code works properly.

**The 1st version will have the following features:**

The ability to move left, right and jump, make sure that the movement is enjoyable and responsive from the start.

Add some objects to the scene to see how they interact as they should. For instance, being able to stand on top/ not walk through.

Add a respawn platform for the player to go to when they die/fall off the map.

| Test No. | Description | Type of test | Test data | Expected Result | Actual Result |
|---|---|---|---|---|---|
| 1 | Can the player move? |  | Input movement button | Player moves | Yes |
| 2 | Does the camera follow the player? |  | Input movement button | Camera moves with the player | Yes |
| 3 | Can the player interact with objects? |  | Place player on top of a platform | Player stands on top | yes |
| 4 | Does the player respawn? |  | Move the player off the map | Player is placed back on its respawn point | Yes |
| 5 | Can the player jump? |  | Input the jump button | Player jumps | Yes |

Video: Stage 1

**Stage 2:**

To make the game more complete, I will add more features to the game, this includes a working UI that includes things such as settings, a start menu and a pause menu during the game. I have chosen these as they are core components of a UI in most

**The 2nd stage will have the following features:**

A menu that appears when the game starts that lets you play the game or enter settings/ the character creator

A pause menu that will let you return to the menu during the game, the game will also freeze when the pause menu appears.

| Test No. | Description | Type of test | Test data | Expected Result | Actual Result |
|---|---|---|---|---|---|
| 1 | Does the game start? |  | Press the start button | Scene, and sprites appear |  |
| 2 | Does a pause menu appear? |  | Press the pause button | Pause menu appears |  |
| 3 | Game stops when paused |  | Press the pause button | Game objects stop moving |  |

| | | | | | |
|---|---|---|---|---|---|
| 4 | Can you navigate the start menu? | | Click each button in the UI | Pages switch when the buttons are pressed | |
| 5 | Does the Controls menu work | | Press buttons to change key binds. | Menu appears to change key | |

Video: Stage 2

**Stage 3:**

To further improve the quality of the movement mechanics, I will add more complex actions that the player can execute. These will include:

A double jump that allows the player to jump again while already in the air, potentially playing an animation, letting the player jump further.

A rope/grappling hook that can be thrown/shot to attach to the ceiling, to let the player swing across a gap that would otherwise be too large to travel across

A bomb that can be dropped to propel the player quickly in a certain direction.

| | | | | | |
|---|---|---|---|---|---|
| 1 | Can the player double jump? | | Press space twice | Player can jump | |
| 2 | Can the rope attach to the Ceiling? | | Touch the two objects | Rope attaches to the object | |
| 3 | Can the player fire the rope? | | Press the action button for the rope | Rope moves until it reaches an object | |
| 4 | Does the bomb propel the player? | | Press the action button for the bomb | Moves the player away from the bomb | |

Video: Stage 3

**Stage 4:**

After adding more complex mechanics to the game, I would like to start making it more playable and fun. This means adding things like enemies and fighting, plus changing features to be higher quality, this includes changes such as bug fixes and quality of life changes like the player’s interactions with objects and other characters.

**The 4th stage will have the following features:**

Enemies that can move towards or fire objects at the player. The player should also be able to take damage from these attacks or by touching an enemy. But, they can fight back using the tools they have been given (bomb/grappling hook) to damage the enemies.

| Test No. | Description | Type of test | Test data | Expected Result | Actual Result |
|---|---|---|---|---|---|
| 1 | Do the enemies try to attack the player? | | Move player towards enemy. | Enemies aggress towards the player, once they get close enough | |
| 2 | When hit by an attack, does the player take damage/get reset? | | Player touches component that damages them | Teleported to the beginning of the screen | |
| 3 | Do the enemies take damage from the player's attacks? | | Enemy touches player attack. | Enemy takes damage/ disappears. | |
| 4 | Is the player able to attack? | | Press attack key | Player attacks | |

Video: Stage 4

**Stage 5:**

Once the base mechanics of the game are functional, I will add levels inside the game for the user to play. These will be simple platforming at the start, that make use of the tools the player has been given after the end of each level.

| Test No. | Description | Type of test | Test data | Expected Result | Actual Result |
|---|---|---|---|---|---|
| 1 | Is the player able to complete each level? | | Player attempts to complete level | It is possible to reach the end of the level | |
| 2 | Does the player receive a new tool at the end of a level? | | Execute function to give the player a new tool | Player receives it in the tool slot. | |
| 3 | Does an image appear to display the unlocked tool to the player. | | Execute function to give the player a new tool | Image appears, showing the new tool. | |

**Stage 6:**

I will continue with the 6th stage when most other parts of the game have been completed, the reason for this is that animations take a long time to complete, so they are a low priority part of the project, as other stages, like movement and level design.

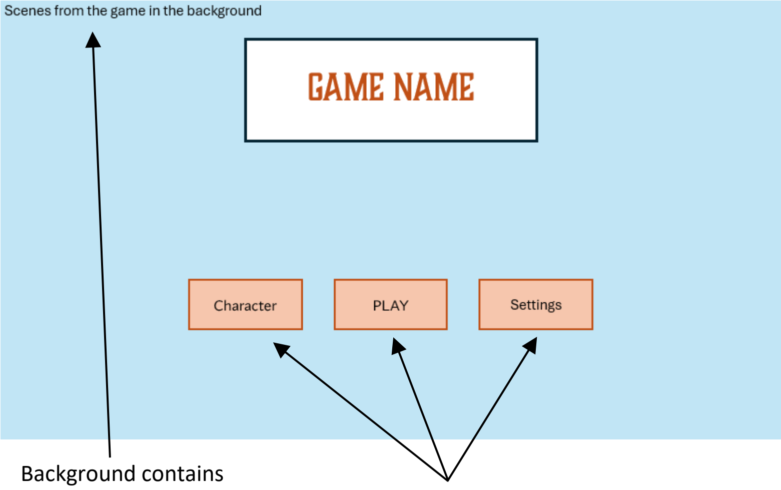
**The 6th stage will have the following features:**

Animations play for player and enemy movement and attacking, this will add more depth to the game as the gameplay as a whole will seem more complete and alive. On top of this, the UI will have simple animations while navigating, this makes the GUI seem more interactive, making it enjoyable to use.

| | | | | | |
|---|---|---|---|---|---|
| 1 | Does an animation play when the enemies move? | | Move the enemy | Movement animation plays | |
| 2 | Does an animation play while the enemy is attacking? | | Execute enemy attack function | Attack animation plays | |
| 3 | Does an animation play when the player moves? | | Press the player movement key | Player animation plays | |
| 4 | Does an animation play when the player attacks? | | Press the player attack key | Attack animation plays | |
| 5 | Do simple animations play when clicking buttons in the UI | | Hover mouse over button | Subtle colour change to show selected button | |
| 5 | Do simple transition animations play when navigating the UI | | Click button to change between screens | Transition animation plays to make | |

Video: Stage 6

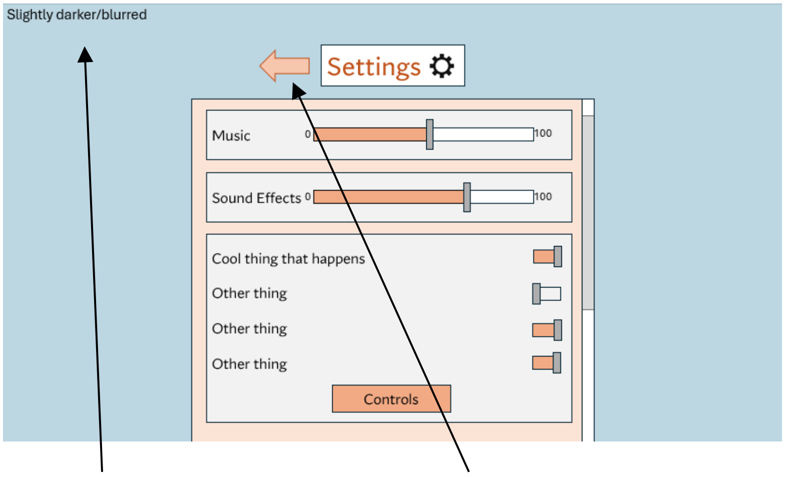
**GUI Design:**



Background contains scenes from the game, so the placeholder colour is blue, as it is abundant due to the game being outside

Each button takes the user to its respective page

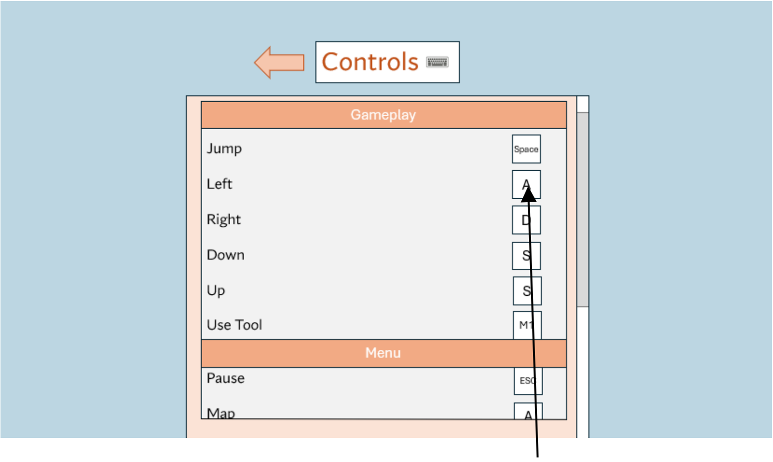
The game will open to this page which contains the title of the game, and multiple buttons below. I have done it in this way as it makes the name more prominent, letting the player know what game it is instantly, as opposed to something else being more noticeable, making the game name less obvious. The buttons are just below the title, in an easy-to-read font that stands out compared to the orange background. I have decided to use the colour orange as it stands out against the blue background, which is blue as it will contain backgrounds from the game, which is mainly set outside, with a blue sky.



The settings background is slightly darker compared to the main menu, the background will also become blurred; I think this helps to indicate that it is a sub-page and take the attention away from the background and focus on the main point of the page

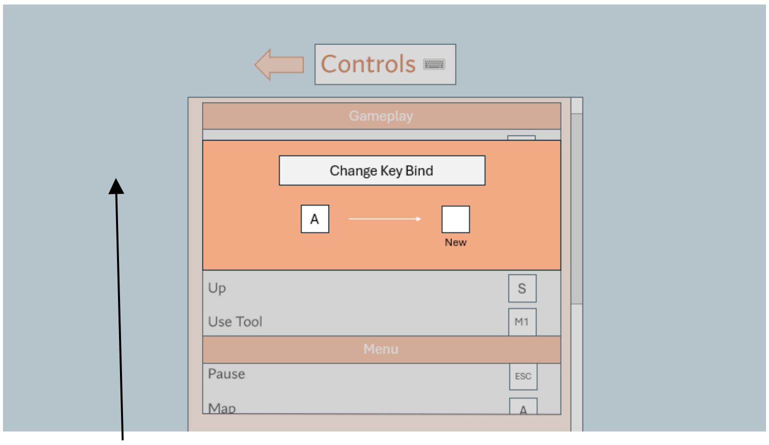
Back arrow indicates that it will bring the user back to the main menu. This is used instead of 'back' as it became too cluttered with words and I preferred it to be more simple

When the settings button of the previous layout is pressed, this page will appear. There is a title at the top of the page telling the user it is the settings, plus an icon, just in case the user isn't able to read English. This tells them they are in the settings. Below the title, there are different sliders and buttons you can press. For the sliders, there are contrasting colours: orange and white, which is a good visual representation to show how far along it is. Similarly, the buttons have that feature, where it is obvious if they are on or not. These are helpful as they make navigating settings as easy as possible.



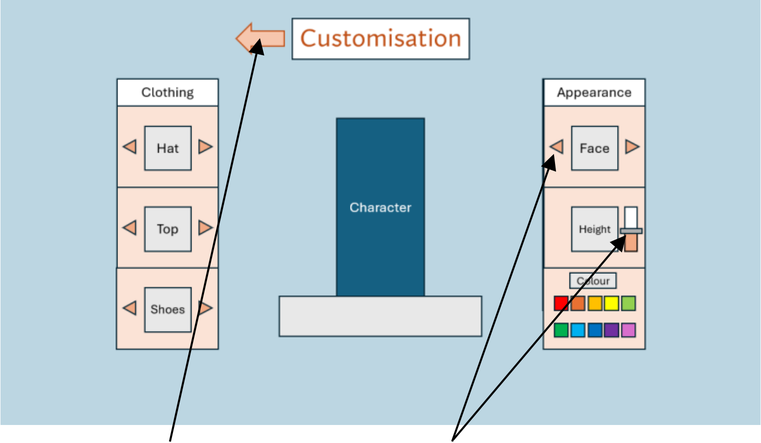
Clicking this will allow the user to change the key the action is bound to

After the Controls button has been pressed, this page appears, it shows the user what each key does in game, this is helpful in case they do not know. The background is a different colour to the key binds' background, this makes them stand out more and easier to see. I have also used a keyboard icon next to the title to help indicate the usage of the page, like the settings icon. The user can also change the keys by clicking on them. There is also a heading that groups the controls into categories, this makes a certain one easier to find as they are grouped together, as opposed to all in a list, which is harder to sort through.



The background is again darker, to indicate what page the game is focused on.

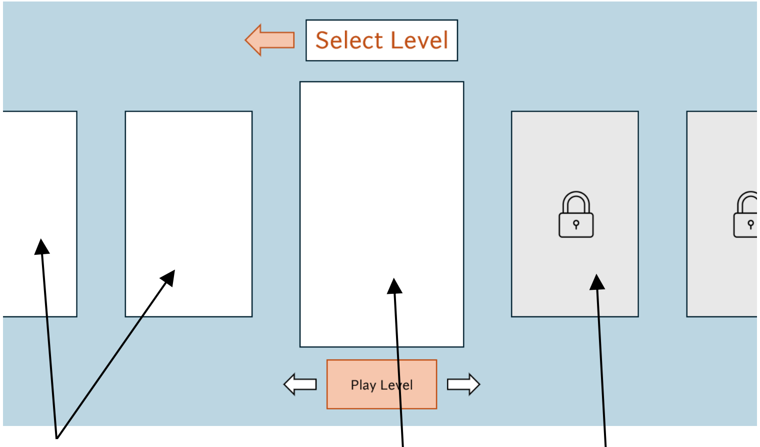
When the user tries to change their key binds, this window will appear, where they can press any button to change the key. The background becomes darker as it takes attention away and places it on the new window. It is orange as it matches with the colour scheme of the other components of the GUI.



Large arrow to indicate that it will take the user back to the main menu.

Allows the user to cycle along the different options/ change the appearance

The Customisation page is clearly labelled at the top, I would like to add an icon, like an item of clothing or something similar. The character is located in the middle of the screen, surrounded by windows that can be used to change the design. On the left, it is clearly labelled 'clothing' and there are arrows on each component that can be used to cycle along to the next one. This Is the same for the 'appearance' window, but the height has a slider. I chose this over something else like a set height/ a few set heights, as it allows the user to fully customize the character. And there are a few select colours to choose from.



Large arrow to indicate that it will take the user back to the main menu.

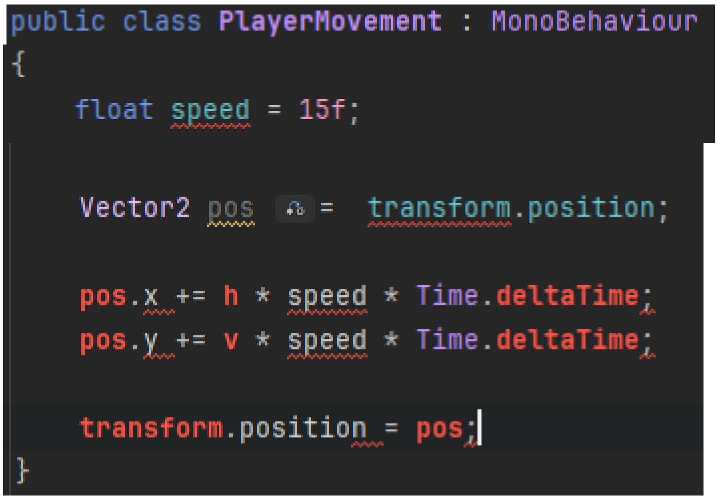
Larger level image to show it's the one selected.

Locked levels have a lock icon covering the image.

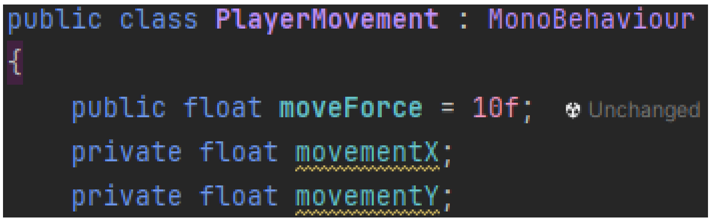
The level select screen contains a list of the different levels the user has unlocked, in the form of multiple different images in a row. Levels that haven't been unlocked are hidden behind a lock icon and potentially a blurry image of the level to be unlocked. There are arrows to the left and right of the 'Play Level' button, which can be used to scroll along the row. When the level is the one that can be selected, it is larger, indicating it's the one the user is on.

**Development:**

Player:

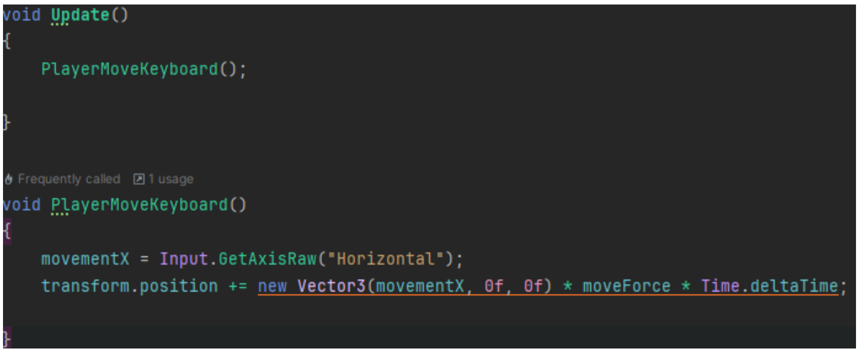


```
public class PlayerMovement : MonoBehaviour
{
    float speed = 15f;

    Vector2 pos  = transform.position;

    pos.x += h * speed * Time.deltaTime;
    pos.y += v * speed * Time.deltaTime;

    transform.position = pos;
}
```



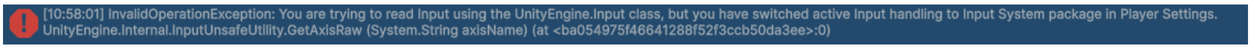
```
public class PlayerMovement : MonoBehaviour
{
    public float moveForce = 10f;  Unchanged
    private float movementX;
    private float movementY;
```

he first thing I wanted to do was make the player move. I started off by creating a variable named speed, which will decide how fast the player moves, I later changed this to moveForce as I think it better describes how it affects the player. At first, I also used the transform function to move the player, which changes the player's position, but later changed it to directly affect the rigidbody of the player, which is better for physics-based movement which I would like. I also declared variables movementX and movementY, these are the directions that the player will move.



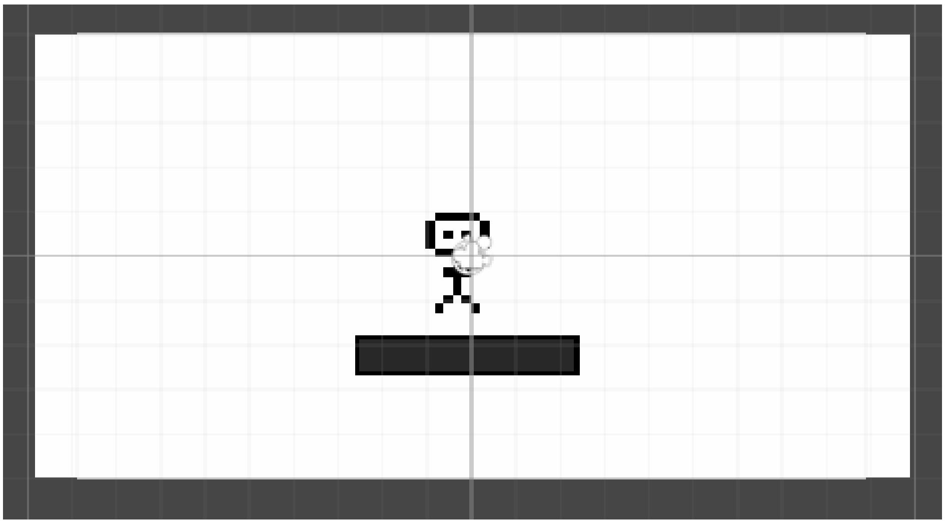
```
void Update()
{
    PlayerMoveKeyboard();

}

 Frequently called   1 usage
void PlayerMoveKeyboard()
{
    movementX = Input.GetAxisRaw("Horizontal");
    transform.position += new Vector3(movementX, 0f, 0f) * moveForce * Time.deltaTime;

}
```

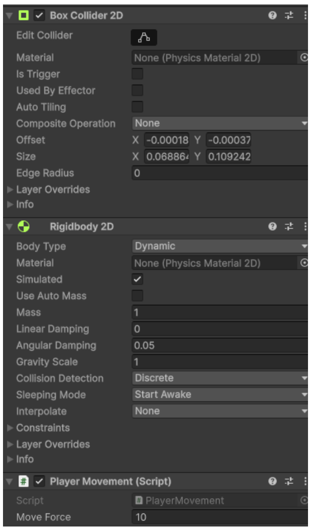
I then made a function which checks what button is pressed on the keyboard in 'GetAxisRaw' and adds it to the transform position equation. This multiplies the x position, moveForce and Time.deltaTime to move the player.



```
[10:58:01] InvalidOperationException: You are trying to read Input using the UnityEngine.Input class, but you have switched active Input handling to Input System package in Player Settings.
UnityEngine.Internal.InputUnsafeUtility.GetAxisRaw (System.String axisName) (at <ba054975f46641288f52f3ccb50da3ee>:0)
```

Program was not running. Fixed by changing 'Active Input Handling' to 'both





☑ Box Collider 2D

| | |
|---|---|
| Edit Collider | |
| Material | None (Physics Material 2D) |
| Is Trigger | □ |
| Used By Effector | □ |
| Auto Tiling | □ |
| Composite Operation | None |
| Offset | X -0.00018 Y -0.00037 |
| Size | X 0.06886 Y 0.109242 |
| Edge Radius | 0 |
| Layer Overrides | |
| Info | |

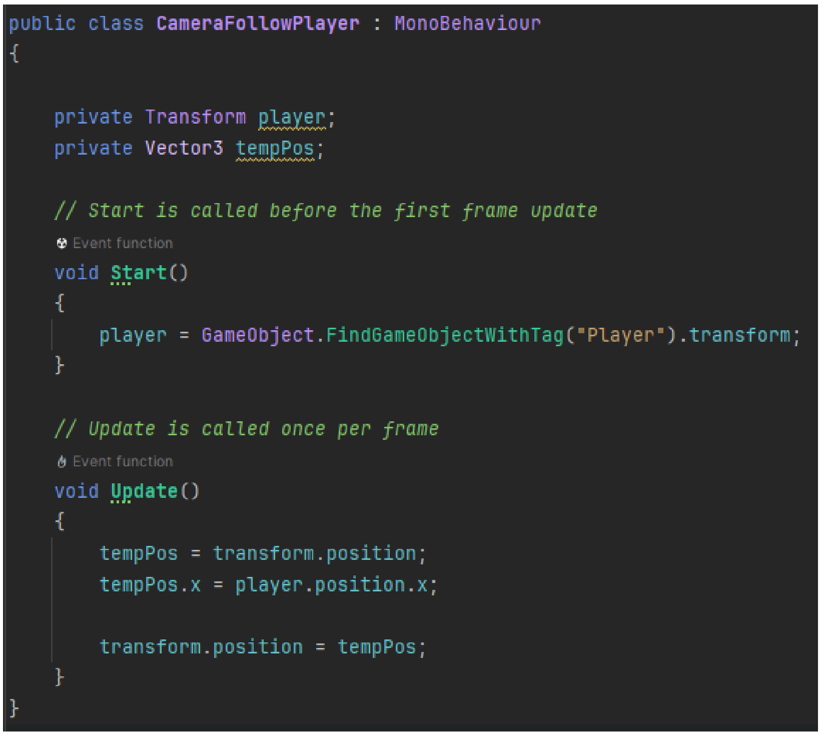
Rigidbody 2D

| | |
|---|---|
| Body Type | Dynamic |
| Material | None (Physics Material 2D) |
| Simulated | ☑ |
| Use Auto Mass | □ |
| Mass | 1 |
| Linear Damping | 0 |
| Angular Damping | 0.05 |
| Gravity Scale | 1 |
| Collision Detection | Discrete |
| Sleeping Mode | Start Awake |
| Interpolate | None |
| Constraints | |
| Layer Overrides | |
| Info | |

☑ Player Movement (Script)

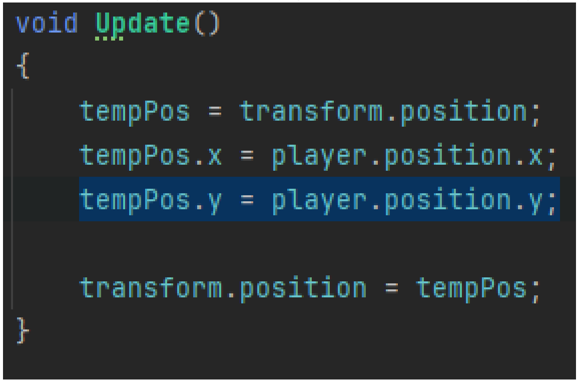
| | |
|---|---|
| Script | PlayerMovement |
| Move Force | 10 |

Camera:



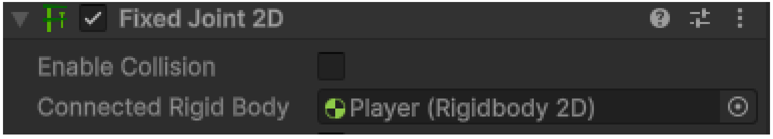
```
public class CameraFollowPlayer : MonoBehaviour
{

    private Transform player;
    private Vector3 tempPos;

    // Start is called before the first frame update
    Event function
    void Start()
    {
        player = GameObject.FindGameObjectWithTag("Player").transform;
    }

    // Update is called once per frame
    Event function
    void Update()
    {
        tempPos = transform.position;
        tempPos.x = player.position.x;

        transform.position = tempPos;
    }
}
```

Does not follow player
This is because the character is not called ‘player’ in unity.
So, I renamed it to player for it to work
Camera only followed player horizontally



```
void Update()
{
    tempPos = transform.position;
    tempPos.x = player.position.x;
    tempPos.y = player.position.y;

    transform.position = tempPos;
}
```

Tools:
Added a placeholder grappling hook that the player could hold. At first, I thought I could attach the player movement script to it and have them move together, but that didn’t work.
(Unity, 2025)



☑ Fixed Joint 2D

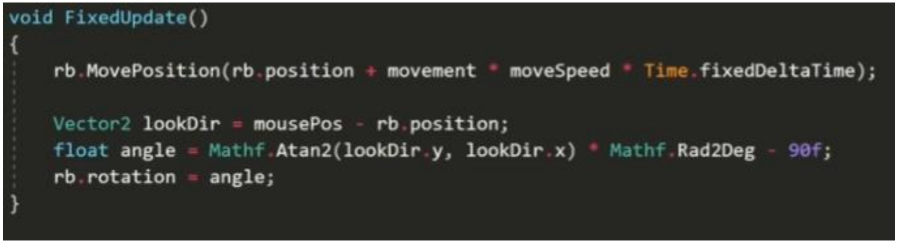
Enable Collision □

Connected Rigid Body: Player (Rigidbody 2D)

I used this to connect the player and the grappling hook

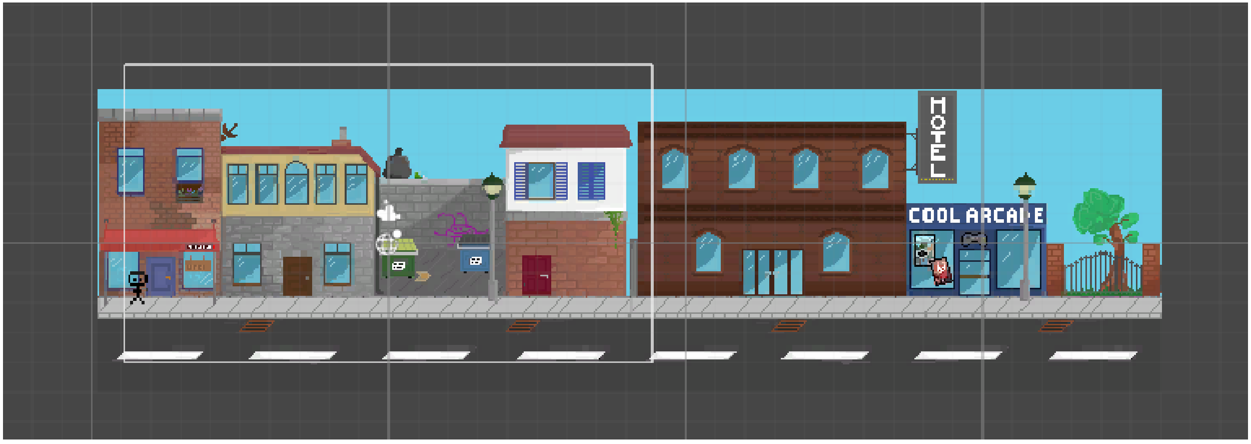
I wanted to have the tool point towards the player’s mouse cursor.
(Brackeys, 2025) – this video on top-down shooting provides code that makes the player look at

the mouse position. I watched that section of the video and used the in-built functions I had learned to add it to my program.

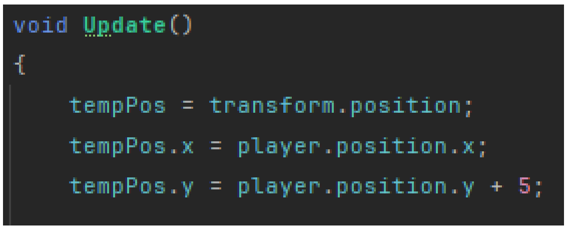


```
void FixedUpdate()
{
    rb.MovePosition(rb.position + movement * moveSpeed * Time.fixedDeltaTime);

    Vector2 lookDir = mousePos - rb.position;
    float angle = Mathf.Atan2(lookDir.y, lookDir.x) * Mathf.Rad2Deg - 90f;
    rb.rotation = angle;
}
```

I then added my textures that I had made, of a city street



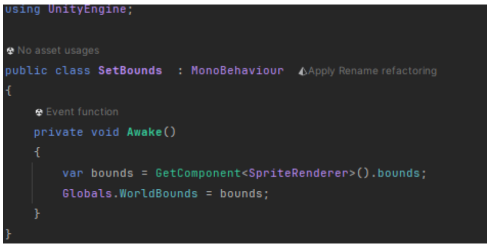
But the camera wasn't focused on the street, and you could see outside, which I didn't want



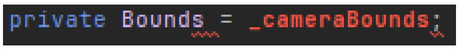
```
void Update()
{
    tempPos = transform.position;
    tempPos.x = player.position.x;
    tempPos.y = player.position.y + 5;
```

I added 5 to the position of the camera to see how that would affect it. I then settled on 5/2 being a good number to use as you could see the whole of the scene.

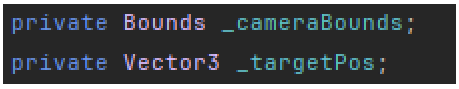
I then wanted the camera the stop when the player reaches the end of the street, I couldn't figure out how to do this, so I watched a tutorial that (chonk, 2025) showed me how, and then I used what I had learned to code it into my game.



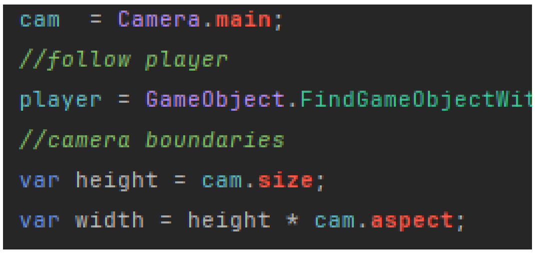
```
using UnityEngine;

No asset usages
public class SetBounds  : MonoBehaviour  Apply Rename refactoring
{
    Event function
    private void Awake()
    {
        var bounds = GetComponent<SpriteRenderer>().bounds;
        Globals.WorldBounds = bounds;
    }
}
```



```
private Bounds = _cameraBounds;
```

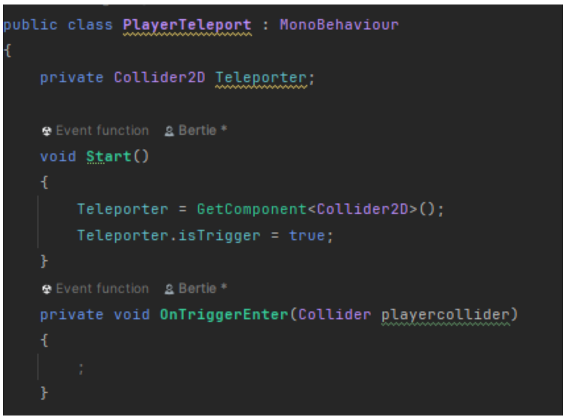


```
private Bounds _cameraBounds;
private Vector3 _targetPos;
```



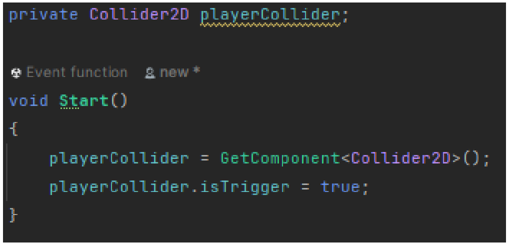
```
cam  = Camera.main;
//follow player
player = GameObject.FindGameObjectWit
//camera boundaries
var height = cam.size;
var width = height * cam.aspect;
```

To move the player between levels, I have decided I want to have them teleport in game to change the scene, to test this, I made 2 portals and had the player teleport between them. I also want the screen to momentarily turn black to make the transition more seamless, as I will use this often, it will be a public function so it can be used all the time. (Unity, 2025) helped me to use the isTrigger function and onTriggerEnter to check if the player was touching the portal.

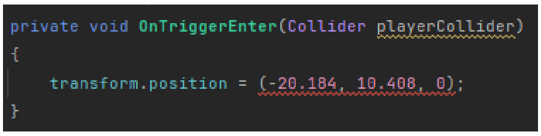


```
public class PlayerTeleport : MonoBehaviour
{
    private Collider2D Teleporter;

    Event function  Bertie *
    void Start()
    {
        Teleporter = GetComponent<Collider2D>();
        Teleporter.isTrigger = true;
    }
    Event function  Bertie *
    private void OnTriggerEnter(Collider playercollider)
    {
        ;
    }
```

I haven't added anything to the function yet as I need to add a collider for the player.

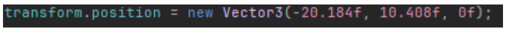


```
private Collider2D playerCollider;

Event function  new *
void Start()
{
    playerCollider = GetComponent<Collider2D>();
    playerCollider.isTrigger = true;
}
```



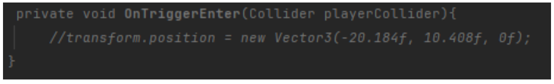
```
private void OnTriggerEnter(Collider playerCollider)
{
    transform.position = (-20.184, 10.408, 0);
}
```

I tried just changing the position of the player when they entered the portal, but that didn't work. I then added an f after each number to specify a float, with a vector3 object. This came back with no errors



```
transform.position = new Vector3(-20.184f, 10.408f, 0f);
```

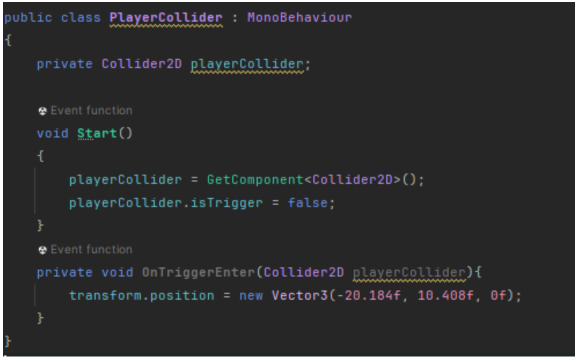
When I started the program, the player kept falling through the floor, I didn't know if it was a problem with the player or the ground, so I made a new object of a 2d box, the box fell and landed on the ground so I knew it was a problem with the player object.



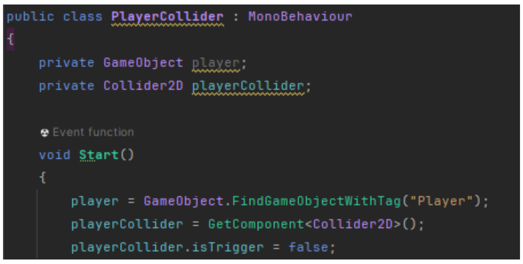
```
private void OnTriggerEnter(Collider playerCollider){
    //transform.position = new Vector3(-20.184f, 10.408f, 0f);
}
```

I temporarily removed the code I had just added and it worked, because the isTrigger box being selected was removing the collision.

After this, the player wasn't being teleported, so I created a new object called PlayerCollider which moved with the player and moved the collider code from the player to a new script attached to that object.

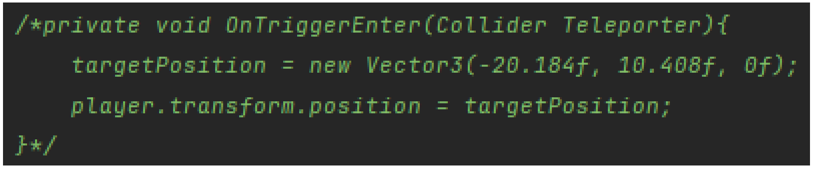


```
public class PlayerCollider : MonoBehaviour
{
    private Collider2D playerCollider;

    Event function
    void Start()
    {
        playerCollider = GetComponent<Collider2D>();
        playerCollider.isTrigger = false;
    }
    Event function
    private void OnTriggerEnter(Collider2D playerCollider){
        transform.position = new Vector3(-20.184f, 10.408f, 0f);
    }
}
```

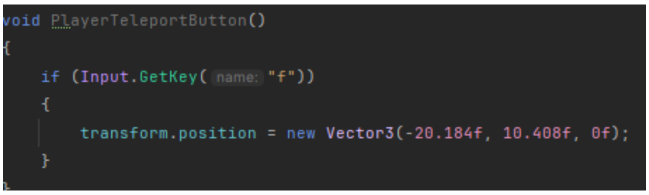


```
public class PlayerCollider : MonoBehaviour
{
    private GameObject player;
    private Collider2D playerCollider;

    Event function
    void Start()
    {
        player = GameObject.FindGameObjectWithTag("Player");
        playerCollider = GetComponent<Collider2D>();
        playerCollider.isTrigger = false;
```

This code ran, but it still didn’t work, so I decided to teleport the player using a button that the player clicks when they get close to the teleporter. I made the previous code a comment.



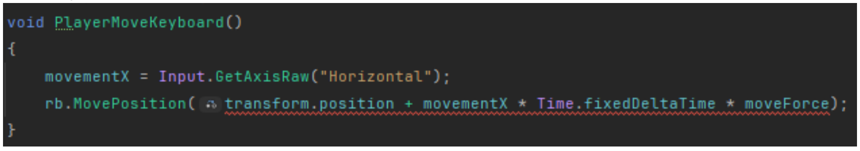
```
/*private void OnTriggerEnter(Collider Teleporter){
    targetPosition = new Vector3(-20.184f, 10.408f, 0f);
    player.transform.position = targetPosition;
}*/
```



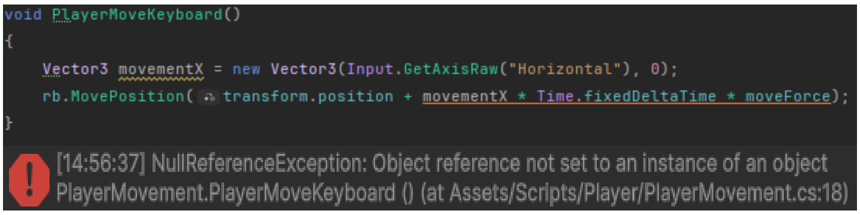
```
void PlayerTeleportButton()
{
    if (Input.GetKey( name: "f"))
    {
        transform.position = new Vector3(-20.184f, 10.408f, 0f);
    }
```

I added this function that just checks for when the ‘F’ key is pressed, I chose F as it is close to the WASD and E controls which are used in the game. And this moved the player to the correct position

Then I added a barrier to stop the player falling off the map, but the player could walk through it, so I changed the movement from transform.position to Rigidbody2d.MovePosition. (Unity, 2025)



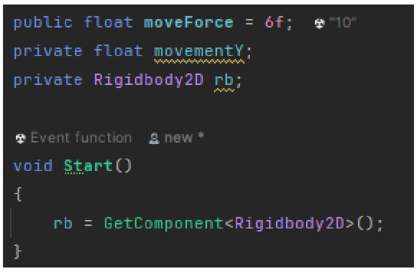
```
void PlayerMoveKeyboard()
{
    movementX = Input.GetAxisRaw("Horizontal");
    rb.MovePosition( transform.position + movementX * Time.fixedDeltaTime * moveForce);
}
```



```
void PlayerMoveKeyboard()
{
    Vector3 movementX = new Vector3(Input.GetAxisRaw("Horizontal"), 0);
    rb.MovePosition( transform.position + movementX * Time.fixedDeltaTime * moveForce);
}
```

```
[14:56:37] NullReferenceException: Object reference not set to an instance of an object
PlayerMovement.PlayerMoveKeyboard () (at Assets/Scripts/Player/PlayerMovement.cs:18)
```

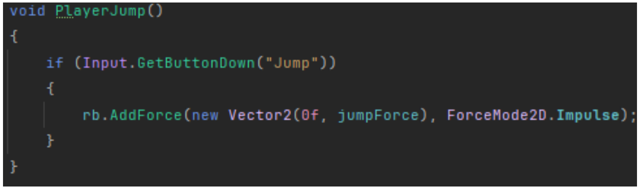
This didn’t work at first, but then I changed movementX to Vector3 and removed the line initializing it as a float. And there were no errors, until I ran the program and there was a continuous error, I didn’t know what was wrong, so I used this site (plbm, 2025) to help me figure out why.



```
public float moveForce = 6f;   "10"
private float movementY;
private Rigidbody2D rb;

Event function   new *
void Start()
{
    rb = GetComponent<Rigidbody2D>();
}
```

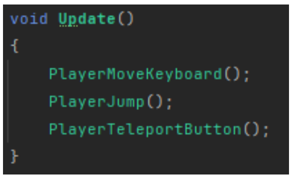
I then added this line inside of the start function which referenced the rigidbody, which applies a force to the rigid body instead of just changing the position with transform. This worked, the player moves and cannot move through the barrier I placed.

I now wanted to make the player jump, as I had sorted out the movement.

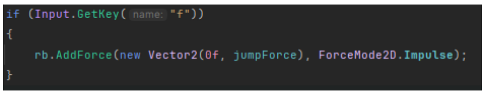


```
void PlayerJump()
{
    if (Input.GetButtonDown("Jump"))
    {
        rb.AddForce(new Vector2(0f, jumpForce), ForceMode2D.Impulse);
    }
}
```

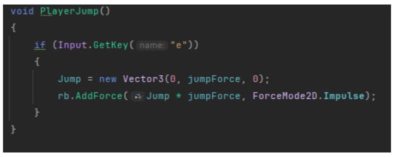
I added a function named PlayerJump to void Update, as that is called every frame, so it will always be checking for a button to be pressed. I added the impulse mode so the force is applied instantly.
This didn’t work at first, as the player didn’t jump when the space bar was pressed. I then changed the input type from GetButtonDown to GetKey and it still didn’t work, as the player only jumped occasionally, and with a lot of



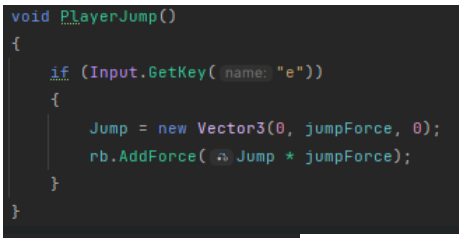
```
void Update()
{
    PlayerMoveKeyboard();
    PlayerJump();
    PlayerTeleportButton();
}
```



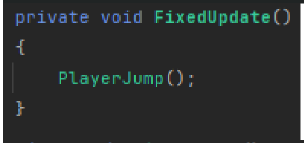
```
if (Input.GetKey( name: "f"))
{
    rb.AddForce(new Vector2(0f, jumpForce), ForceMode2D.Impulse);
}
```



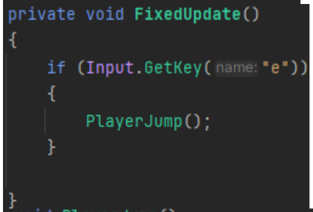
```
void PlayerJump()
{
    if (Input.GetKey( name: "e"))
    {
        Jump = new Vector3(0, jumpForce, 0);
        rb.AddForce( Jump * jumpForce, ForceMode2D.Impulse);
    }
}
```



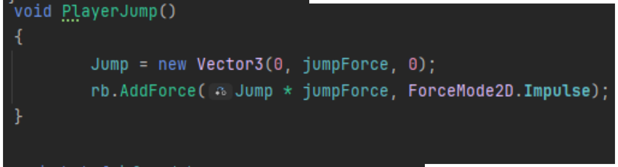
```
void PlayerJump()
{
    if (Input.GetKey( name: "e"))
    {
        Jump = new Vector3(0, jumpForce, 0);
        rb.AddForce( Jump * jumpForce);
    }
}
```



```
private void FixedUpdate()
{
    PlayerJump();
}
```



```
private void FixedUpdate()
{
    if (Input.GetKey( name: "e"))
    {
        PlayerJump();
    }

}
```

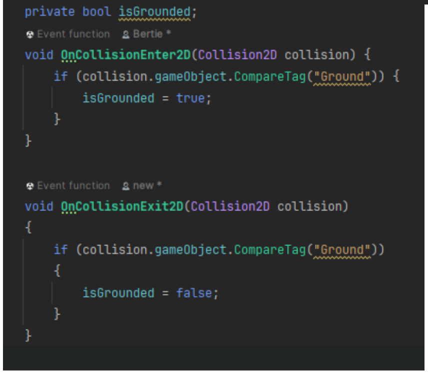


```
void PlayerJump()
{
        Jump = new Vector3(0, jumpForce, 0);
        rb.AddForce( Jump * jumpForce, ForceMode2D.Impulse);
}
```

I then removed the Impulse from the line, and the player now jumped but it was quite simple and didn't work very well, I wanted to add it back in to make it seem more like a proper jump. The player also fell slowly so I increased the gravity scale to about 30, as that seemed to mimic real life gravity more accurately.

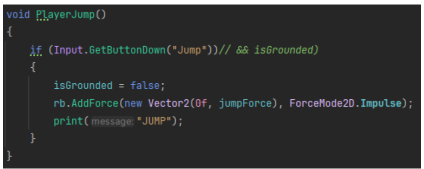
I then moved the player jump to FixedUpdate, which is better for physics-based functions than just the Update function. But the player still didn't jump when the button was pressed.

I then decided to take the inputs for the movement functions in Update rather than in their specific functions, as I thought it may change something.

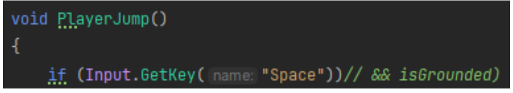


```
private bool isGrounded;
Event function   Bertie *
void OnCollisionEnter2D(Collision2D collision) {
    if (collision.gameObject.CompareTag("Ground")) {
        isGrounded = true;
    }
}

Event function   new *
void OnCollisionExit2D(Collision2D collision)
{
    if (collision.gameObject.CompareTag("Ground"))
    {
        isGrounded = false;
    }
}
```

I decided to move on as it still wasn't jumping, I added 2 functions to see if the player was on the ground and to return true or false, I then edited the jump function to require the player to be touching an object with the 'Ground' tag for it to execute the function.



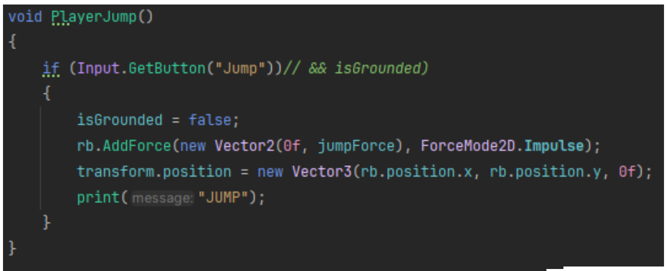
```
void PlayerJump()
{
    if (Input.GetButtonDown("Jump"))// && isGrounded)
    {
        isGrounded = false;
        rb.AddForce(new Vector2(0f, jumpForce), ForceMode2D.Impulse);
        print( message: "JUMP");
    }
}
```



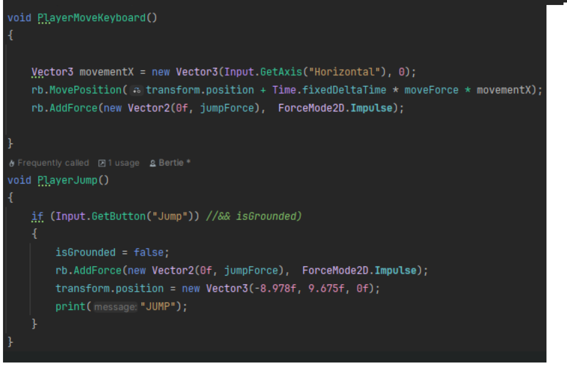
```
void PlayerJump()
{
    if (Input.GetKey( name: "Space"))// && isGrounded)
```

I now focused on making the player jump work properly. First, I added a line that prints 'JUMP' whenever the spacebar is pressed, this is to test whether the input is thing not working or something else.

When I pressed space, the word was output into the console, indicating the problem is with the line where the force is added. I noticed that when I teleport the player (apply transform.position) after pressing jump, it gains a lot of momentum. I thought that I could use this and apply transform.position to the rigidbody of the player but not actually change the location. I did this because it could influence the player and make it jump. This didn't work

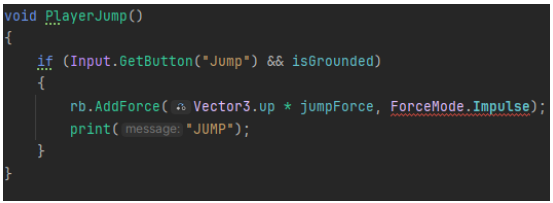


```
void PlayerJump()
{
    if (Input.GetButton("Jump"))// && isGrounded)
    {
        isGrounded = false;
        rb.AddForce(new Vector2(0f, jumpForce), ForceMode2D.Impulse);
        transform.position = new Vector3(rb.position.x, rb.position.y, 0f);
        print( message: "JUMP");
    }
}
```

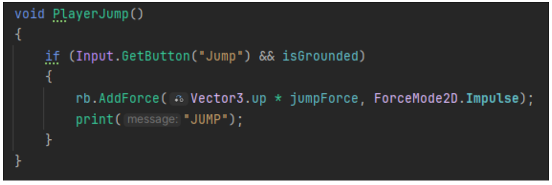


```
void PlayerMoveKeyboard()
{

    Vector3 movementX = new Vector3(Input.GetAxis("Horizontal"), 0);
    rb.MovePosition( transform.position + Time.fixedDeltaTime * moveForce * movementX);
    rb.AddForce(new Vector2(0f, jumpForce),  ForceMode2D.Impulse);

}
Frequently called   1 usage   Bertie *
void PlayerJump()
{
    if (Input.GetButton("Jump")) //&& isGrounded)
    {
        isGrounded = false;
        rb.AddForce(new Vector2(0f, jumpForce),  ForceMode2D.Impulse);
        transform.position = new Vector3(-8.978f, 9.675f, 0f);
        print( message: "JUMP");
    }
}
```

I now have no idea what's wrong. I then removed the addforce line and implemented it into the move function, when the player moved, it also jumped., and when I pressed space, the player successfully teleported. I still had no idea what was wrong. It turns out the function).

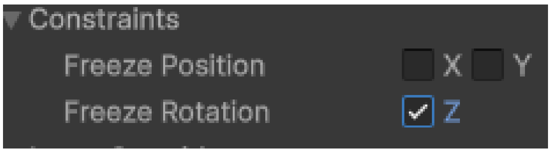


```
void PlayerJump()
{
    if (Input.GetButton("Jump") && isGrounded)
    {
        rb.AddForce( Vector3.up * jumpForce, ForceMode.Impulse);
        print( message: "JUMP");
    }
}
```



```
void PlayerJump()
{
    if (Input.GetButton("Jump") && isGrounded)
    {
        rb.AddForce( Vector3.up * jumpForce, ForceMode2D.Impulse);
        print( message: "JUMP");
    }
}
```

This again, only worked once for the first few seconds, so the player was able to jump one time. To try and fix this issue, I then compared this to other movement code I had written, the only difference was that the horizontal movement was done using transform.position and not applying the force to the rigidbody. I changed my movement code to that, and the player could finally jump. But this is annoying as the player can walk through walls again, as the problem I had previously has now returned. I decided to continue the project and return to this problem at a later stage.



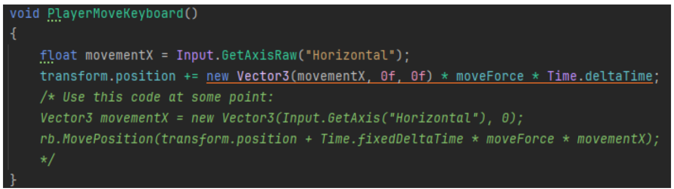
Constraints

Freeze Position □ X □ Y

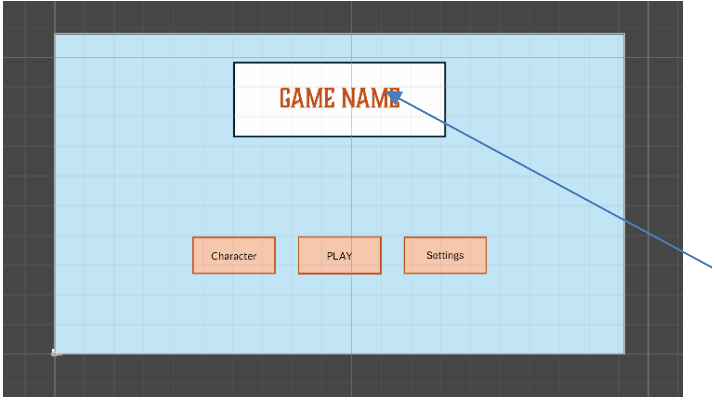
Freeze Rotation ☑ Z

Before I started with stage 2, I wanted to fix a problem that kept appearing with the player, where it was rotating and often falling over, which I didn't want to happen. This was simple as it required selecting a box in the rigidbody editor.

## STAGE 2:

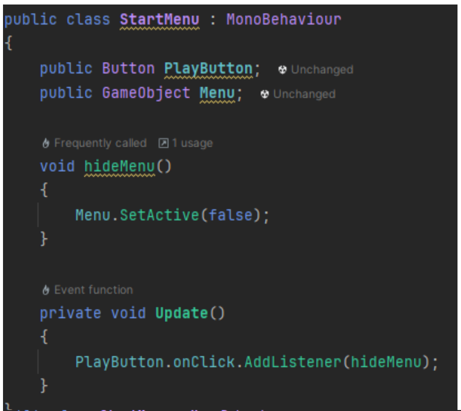


```
void PlayerMoveKeyboard()
{
    float movementX = Input.GetAxisRaw("Horizontal");
    transform.position += new Vector3(movementX, 0f, 0f) * moveForce * Time.deltaTime;
    /* Use this code at some point:
    Vector3 movementX = new Vector3(Input.GetAxis("Horizontal"), 0);
    rb.MovePosition(transform.position + Time.fixedDeltaTime * moveForce * movementX);
    */
}
```



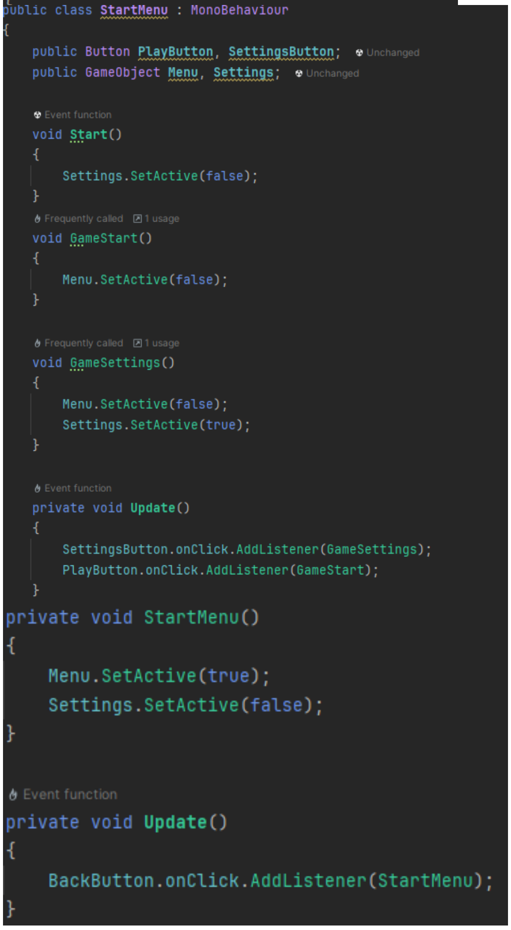
Now that stage 1 was complete, I began with stage 2, where I add a menu screen. At first, I used the GUI design but later I will make a proper one.

I had the idea of just adding sprites that would execute operations when clicked, have the player stand behind the menu screen at first and move position when play was clicked. I realized that was much less efficient and worse than using a canvas, which places the UI elements on screen and allows you to add buttons, which is what I did.



```
public class StartMenu : MonoBehaviour
{
    public Button PlayButton;  Unchanged
    public GameObject Menu;  Unchanged

    Frequently called  1 usage
    void hideMenu()
    {
        Menu.SetActive(false);
    }

    Event function
    private void Update()
    {
        PlayButton.onClick.AddListener(hideMenu);
    }
}
```

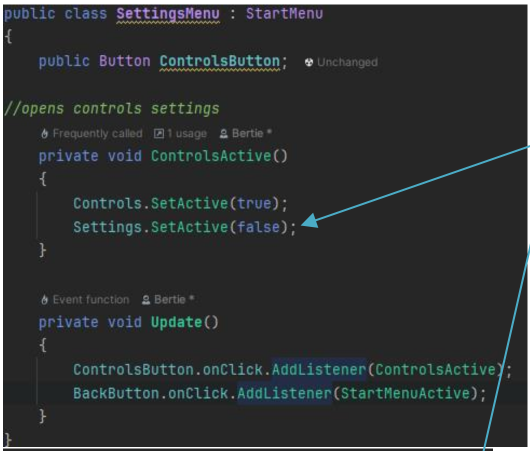
Following the plan for stage 2, I wanted the menu to disappear when the play button was clicked, and for the game to start. This included making sure the player couldn't move until the Play button was pressed. I added the onclick function for the play button, which will set the menu to inactive, which will hopefully make it disappear. This came back with a NullReferenceException, this was because I had not selected the objects that the PlayButton and Menu objects were supposed to reference. This fixed the problem, and the start button works.



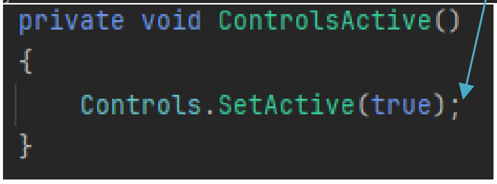
```
public class StartMenu : MonoBehaviour
{
    public Button PlayButton, SettingsButton;  Unchanged
    public GameObject Menu, Settings;  Unchanged

    Event function
    void Start()
    {
        Settings.SetActive(false);
    }
    Frequently called  1 usage
    void GameStart()
    {
        Menu.SetActive(false);
    }

    Frequently called  1 usage
    void GameSettings()
    {
        Menu.SetActive(false);
        Settings.SetActive(true);
    }

    Event function
    private void Update()
    {
        SettingsButton.onClick.AddListener(GameSettings);
        PlayButton.onClick.AddListener(GameStart);
    }
private void StartMenu()
{
    Menu.SetActive(true);
    Settings.SetActive(false);
}

Event function
private void Update()
{
    BackButton.onClick.AddListener(StartMenu);
}
```

As this was now working, I continue to move through the stage and add the settings menu. This currently won't do anything as there's nothing to change now (like key binds and volume), but it will be helpful later in development when I add those things. I changed the function names to make them have a similar naming convention – "GameStart" and "GameSettings" so they are obviously related and part of the same class in the program. This code worked and changed the active panel to settings. But it was slightly transparent for some reason, this was fixed by changing the game object from a panel to an image. The method I have used, where I hide one panel and make another visible, may get confusing and difficult to handle. Because of this, I researched better ways to do this (Unity, 2025) and decided to have each different menu screen in its own canvas, and switch between those instead of having them all in the same one.

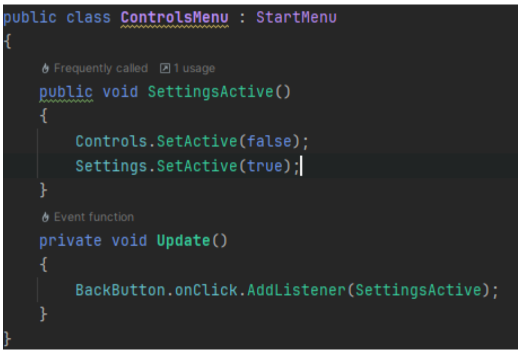
In a new class, I repeated what I did for the previous UI panel, but for settings instead. As I had done a lot of repeated code, I decided to make the Settings class inherit from menu, this made sense to me as they are both part of the UI and as I had already declared the classes and objects in the start menu, I didn't want to repeat that again in the settings one. This all worked as expected, but when the program started, the first panel to appear was the settings menu, when I would like the start screen. As I continued building the UI, a problem occurred where the next panel that I had made wasn't being set to active. While trying to fix this, I removed the line of code that set the settings panel to false. In doing this, the controls one now appeared.



```
public class SettingsMenu : StartMenu
{
    public Button ControlsButton;  Unchanged

//opens controls settings
    Frequently called  1 usage  Bertie *
    private void ControlsActive()
    {
        Controls.SetActive(true);
        Settings.SetActive(false);
    }

    Event function  Bertie *
    private void Update()
    {
        ControlsButton.onClick.AddListener(ControlsActive);
        BackButton.onClick.AddListener(StartMenuActive);
    }
}
```

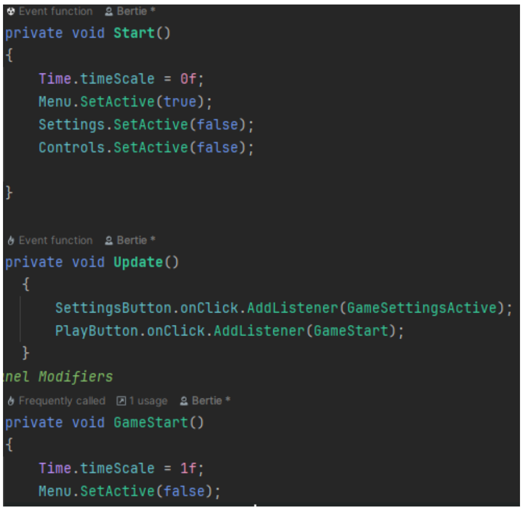


```
private void ControlsActive()
{
    Controls.SetActive(true);
}
```



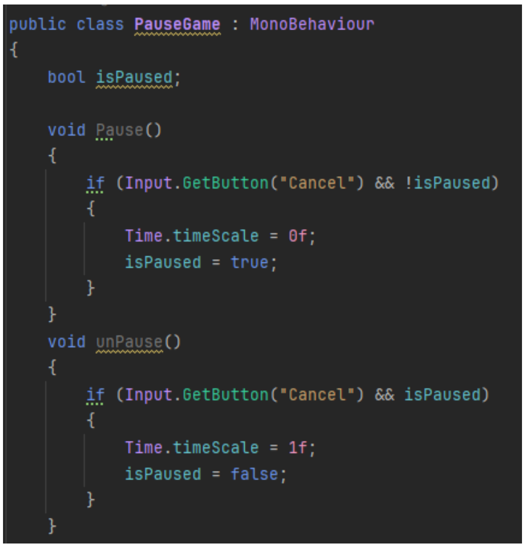
```
public class ControlsMenu : StartMenu
{
    public void SettingsActive()
    {
        Controls.SetActive(false);
        Settings.SetActive(true);
    }
    private void Update()
    {
        BackButton.onClick.AddListener(SettingsActive);
    }
}
```

This is the code for the controls menu, where all it has is a button for going back to settings, and a few placeholder buttons for the future when I add functionality to the screen

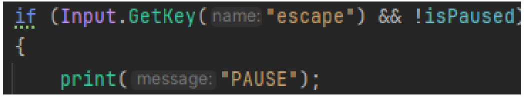


```
private void Start()
{
    Time.timeScale = 0f;
    Menu.SetActive(true);
    Settings.SetActive(false);
    Controls.SetActive(false);

}

private void Update()
  {
      SettingsButton.onClick.AddListener(GameSettingsActive);
      PlayButton.onClick.AddListener(GameStart);
  }
nel Modifiers
private void GameStart()
{
    Time.timeScale = 1f;
    Menu.SetActive(false);
```

My next task was to pause the game until the ‘Play’ button had been pressed. To do this I used the Time.timeScale = 0 method, this will pause physics and time-based operations. I used this method as it is the simplest one for this game, and right now the only movement is the player, so the only thing being paused would be them. Although when enemies and other moving objects are added, I expect this to also work for them.



```
public class PauseGame : MonoBehaviour
{
    bool isPaused;

    void Pause()
    {
        if (Input.GetButton("Cancel") && !isPaused)
        {
            Time.timeScale = 0f;
            isPaused = true;
        }
    }
    void unPause()
    {
        if (Input.GetButton("Cancel") && isPaused)
        {
            Time.timeScale = 1f;
            isPaused = false;
        }
    }
```



```
if (Input.GetKey( name: "escape") && !isPaused)
{
    print( message: "PAUSE");
```

Building on from this, I added a pause button for in-game. I used the timeScale = 0 to stop the player moving when the esc key is pressed. To do this, I created a Boolean to check If the game was paused, which would switch when the game is paused/unpaused. I made use of my previous movement code which makes use of a similar format for checking If the player is on the ground. As this didn’t work, I changed the input method from GetButton to GetKey and printed ‘PAUSE’ into the console to check if the function was executing. I did this because I had done it previously to fix the jump code, and it worked well. This was helpful as when I pressed escape, nothing was printed, helping me pinpoint the location of the error.