Aaron Lee

Professor Eric May

CPSC 386-02

2 December 2023

CPSC 386 assignment/project 3

References used:

- https://kanekizlf.itch.io/jump-sounds
    - Jumping sounds
- https://www.1001fonts.com/karmatic-arcade-font.html
    - Font for titles
- https://wallpapers.com/wallpapers/8-bit-purple-sunset-ijdhaz9e09mteuvg.html
    - Ending screen image
- Unity scripting api:
    - Specifically:
    - https://docs.unity3d.com/ScriptReference/Quaternion.html
    - https://docs.unity3d.com/ScriptReference/Quaternion-eulerAngles.html
    - https://docs.unity3d.com/ScriptReference/Rigidbody2D-gravityScale.html

Instructions for a player…

The game has text that is place on world space that tells the player the controls and what to do with them. In one level I've added an arrow pointing upwards as well with text to let the player know they must use their ability to go up.

Win/loss condition:

If player dies to a spike or water there is a pop up text that states the player is dead and has a message about pressing the key R to restart

For win conditions when player finishes all 5 levels there is an end screen with a different colored background.

Aaron Lee

Professor Eric May

CPSC 386-02

2 December 2023

Interesting decision:

For this one I’ve decided to put a part at the end of the level where a player can use either element to choose what path to take, however only one path is correct, but if they’re fast enough they will not die from their choices (didn’t want to make it unfair).

New Scripts/components:

Button.cs has a new quit command for a button when clicked closes the program:

```
    public void quit() {
        Application.Quit();
    }
}
```

Darkmode.cs is a slider that changes an images opacity to simulate brightness settings.

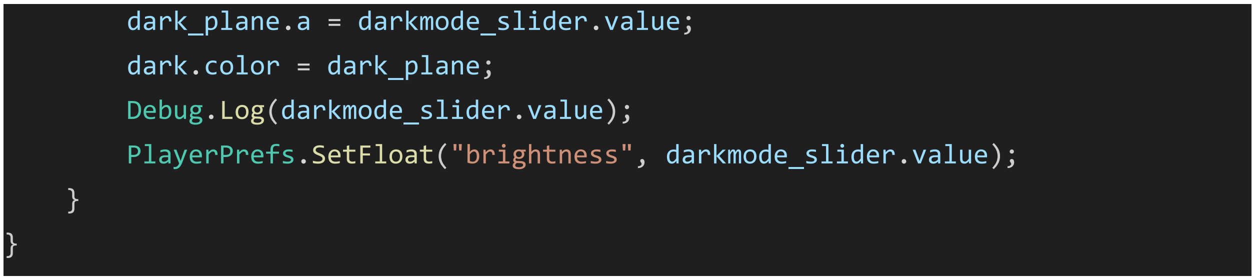
```
public class darkmode : MonoBehaviour
{
    public Image dark;
    public Slider darkmode_slider;
    void Start(){
        GameObject darkmode = GameObject.Find("darkmode");
        float brightness = PlayerPrefs.GetFloat("brightness");
        darkmode_slider.value = brightness;
    }
    void Update(){
        GameObject darkmode = GameObject.Find("darkmode");
        if(darkmode != null){
            brightness();
        }
    }
    public void brightness(){
        GameObject darkmode = GameObject.Find("darkmode");
        Image dark = darkmode.GetComponent<Image>();

        Color dark_plane = dark.color;
```

Aaron Lee

Professor Eric May

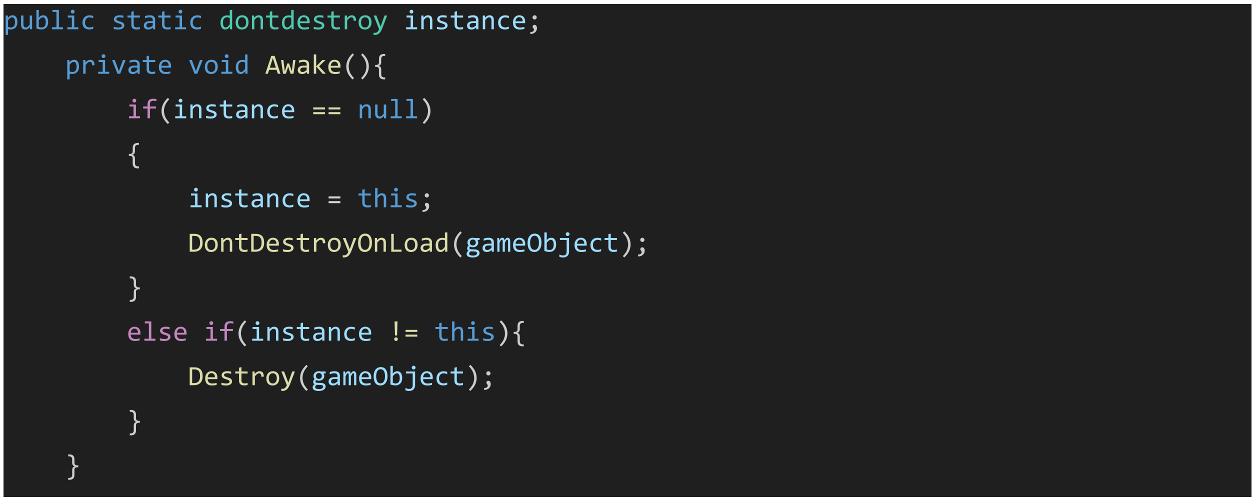
CPSC 386-02

2 December 2023



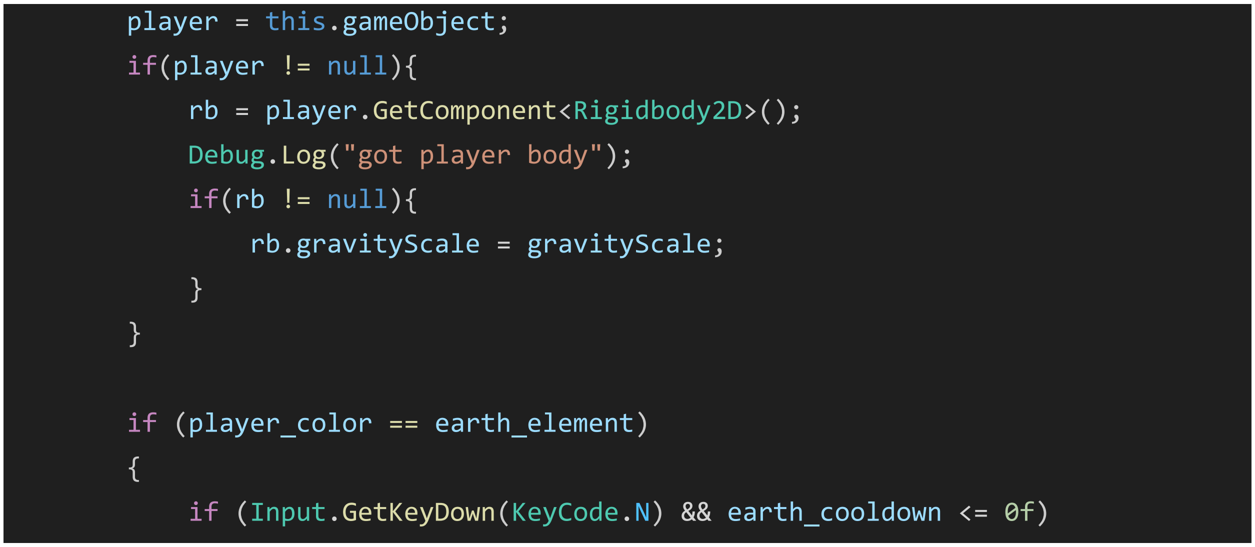
```
        dark_plane.a = darkmode_slider.value;
        dark.color = dark_plane;
        Debug.Log(darkmode_slider.value);
        PlayerPrefs.SetFloat("brightness", darkmode_slider.value);
    }
}
```

Dontdestroy.cs is just to stop darkmode from being destroyed and not destroying the original cause it causes the slider to bug out.



```
public static dontdestroy instance;
    private void Awake(){
        if(instance == null)
        {
            instance = this;
            DontDestroyOnLoad(gameObject);
        }
        else if(instance != this){
            Destroy(gameObject);
        }
    }
```

Added earth_element power to switch gravity (didn’t know what to put for earth so i just thought of being able to pull towards the ceiling and walk upside down).



```
        player = this.gameObject;
        if(player != null){
            rb = player.GetComponent<Rigidbody2D>();
            Debug.Log("got player body");
            if(rb != null){
                rb.gravityScale = gravityScale;
            }
        }

        if (player_color == earth_element)
        {
            if (Input.GetKeyDown(KeyCode.N) && earth_cooldown <= 0f)
```

Aaron Lee

Professor Eric May

CPSC 386-02

2 December 2023

```
            {
                //rb.gravityScale *= -1;
                Debug.Log("n was pressed");
                if(Mathf.Approximately(rb.gravityScale, -0.5f))
                {
                    rb.gravityScale = 0.5f;
                    player.transform.rotation = Quaternion.Euler(0f, 0f, 0f);
                    Debug.Log("Gravity Scale: " + rb.gravityScale);
                    Debug.Log("Rotation: " +
player.transform.rotation.eulerAngles.z);
                    earth_cooldown = 4.0f;
                }
                else if (Mathf.Approximately(rb.gravityScale, 0.5f))
                {
                    rb.gravityScale = -0.5f;
                    player.transform.rotation = Quaternion.Euler(0f, 0f, 180f);
                    Debug.Log("Gravity Scale: " + rb.gravityScale);
                    Debug.Log("Rotation: " +
player.transform.rotation.eulerAngles.z);
                    earth_cooldown = 4.0f;
                }
            }
        }
```

Changed reloader script to reload current active screen

```
        if (Input.GetKeyDown(KeyCode.R))
        {
            Debug.Log("reloaded");
            active_scene = SceneManager.GetActiveScene().name;
            SceneManager.LoadSceneAsync(active_scene);
        }
```

Used textmeshpro to import new font ka1sdf to use in title and ending:

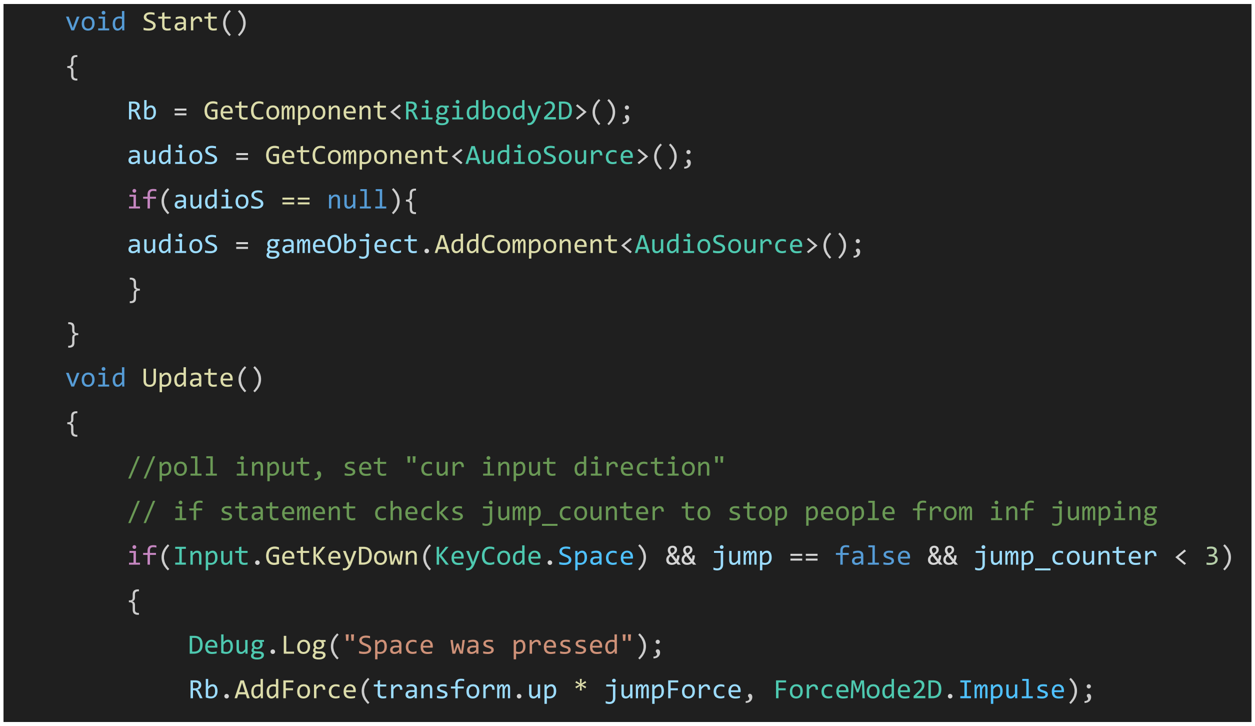
Aaron Lee

Professor Eric May

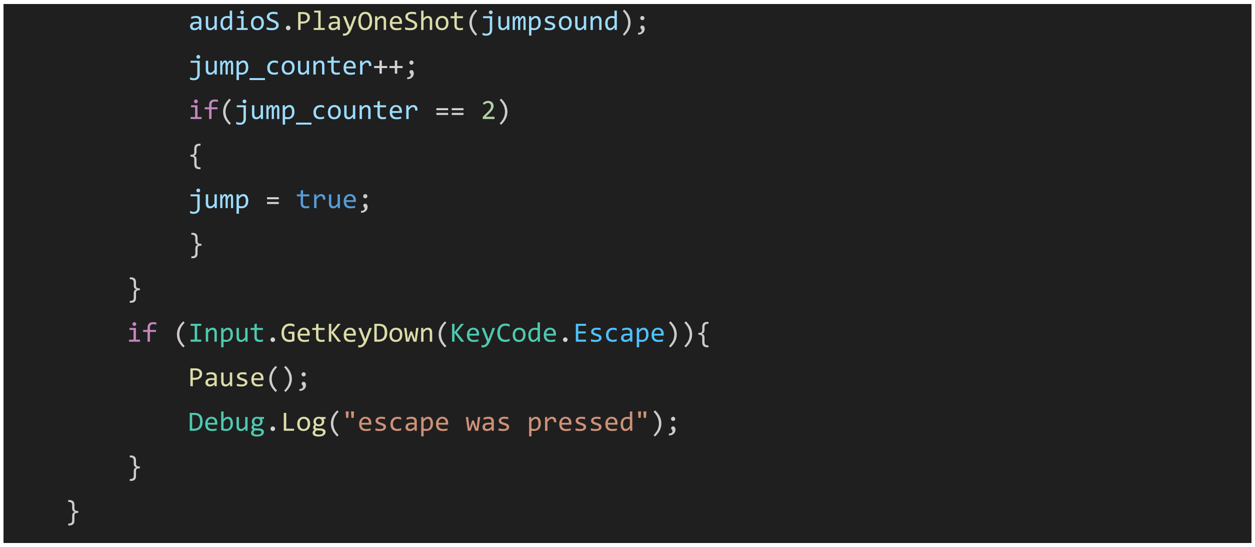
CPSC 386-02

2 December 2023

Added jump sound effects to player in movement.cs



```
    void Start()
    {
        Rb = GetComponent<Rigidbody2D>();
        audioS = GetComponent<AudioSource>();
        if(audioS == null){
        audioS = gameObject.AddComponent<AudioSource>();
        }
    }
    void Update()
    {
        //poll input, set "cur input direction"
        // if statement checks jump_counter to stop people from inf jumping
        if(Input.GetKeyDown(KeyCode.Space) && jump == false && jump_counter < 3)
        {
            Debug.Log("Space was pressed");
            Rb.AddForce(transform.up * jumpForce, ForceMode2D.Impulse);
```



```
        audioS.PlayOneShot(jumpsound);
        jump_counter++;
        if(jump_counter == 2)
        {
        jump = true;
        }
    }
    if (Input.GetKeyDown(KeyCode.Escape)){
        Pause();
        Debug.Log("escape was pressed");
    }
}
```

Created new scenes level 5 and end
Level 5 has gameobjects:

- Player
    - gameobject
    - camera
- Grid
    - Flammable wall (uses wall script to burn on contact)
    - Flammable wall 1 (uses wall script to burn on contact)
    - Ground
    - Spikes
    - Water
    - Deepwater
    - Spikes 2
    - Portal
- Deadtext
- Fire_bullet prefab

Settings:

Added brightness setting that is explained in components

Aaron Lee

Professor Eric May

CPSC 386-02

2 December 2023

Scenes:

Level 5 scene and end scene were created
Main and end screen have exit button to close application

Emergent gameplay:

Level 5 uses mostly everything the player has learned about and creates a large level without instructions to see how a player does.

Key interactions:

1. If a fireball collides with flammable wall it will destroy said wall
2. If a player touches a spike they die
3. If a player is in fire_element or (color red) they and touch water or deeper water they die
4. Ice has no effect on player besides friction being decreased
5. Deepwater should only be accessible by water element. (might need to jump to activate)
6. If player is not touching ground (the ground, dirt, grass with dirt) they will not get a jump reset
7. Player can only jump 1 time on ice
8. Earth_element power reverse gravity and rotates camera to help player match that perspective
9. Earth and fire element have cooldowns of around 3-5 seconds

Problems experienced with project:

1. Difficulty getting the brightness image to stay as the original because script kept deleting the original causing slider to output inf amount of errors.
2. Reversing gravity and rotation cause significant problems as it would not work or not rotate camera. (it works now though)
3. Deepwater and portal disappear when reversing gravity and reappear on normal gravity
4. Deepwater has to have an if statement or it causes other scripts to break.