C# and UI

Sebastian Grygorczuk - STEM Institute at CCNY

Today's Agenda

We're going to be going to be going over Chapter 14 UI Elements

- Learn how to create UI that informs player of their status
- Use Buttons to take player input
- Save Data and Move it Between Scenes

Examples of UI

UI consists of everything that's not directly in the game world.

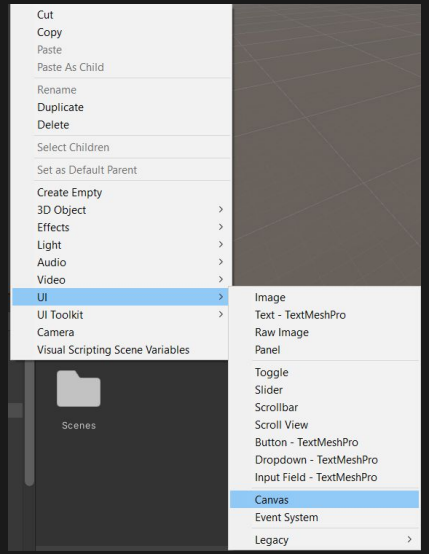
Character Inventory, Overlays that tells you your stats, menus that pause game, title screens, character dialogue section. All of those are UI elements that allow you to convey the general mechanics and ideas to the player.

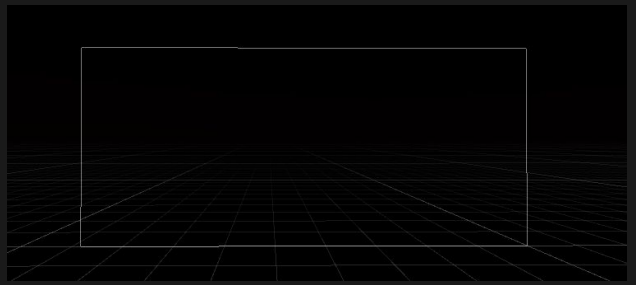
Canvas

To start creating you UI you have to create a Canvas. All UI Game Objects have to be stored under the Canvas Game Object otherwise they will not function.

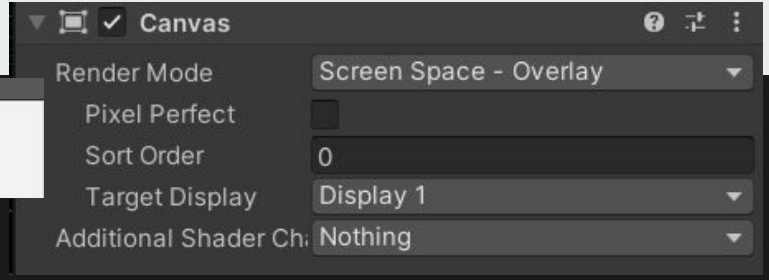
Canvas is the 2D space where the Objects are displayed and is directly linked to how the Camera views the UI elements.

When you create a Canvas you will notice that is is massive, it's because it emulating the size of the screen you are using.





Canvas Component

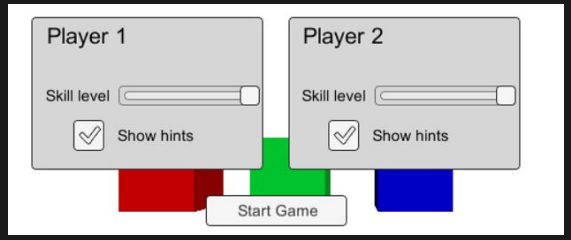


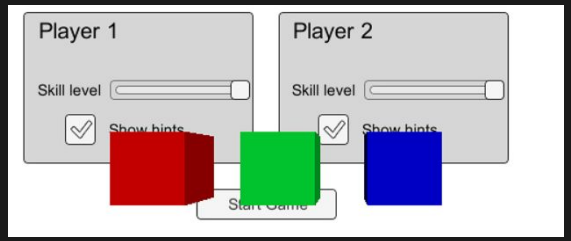
The Canvas component controls where in the Game World the UI should be placed. You have three Options. Screen Space -Overlay, Screen Space - Camera, and World Space.

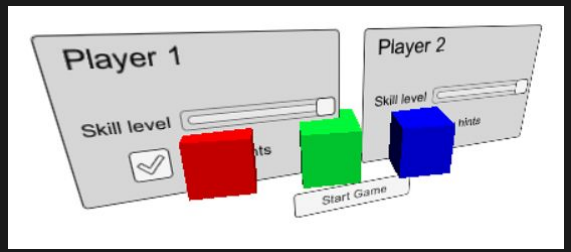
The Screen Space tells you that the UI element will always be attached to the Screen at the same position and size all the time.

Overlay will always keep the UI elements on top of the game while Camera will allow objects to clip into the UI if the Camera moves to them.

World Space on the other hand tells the UI that it exists away from the Camera and will be different size depending on the distance of the player to it.





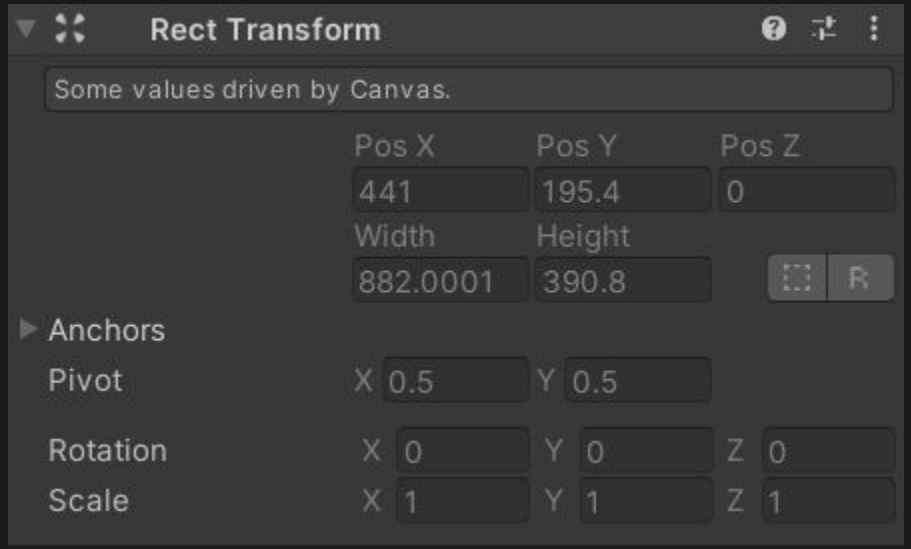


Sebastian Grygorczuk - STEM Institute at CCNY

Rect Transform

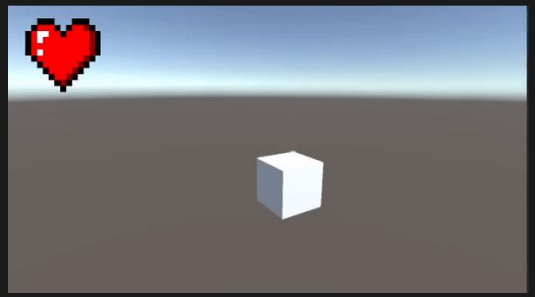
Rect Transform works similar to regular transform with the exception that when you have Overlay or Camera setting on you don't get to control it, it's decided by the camera position and the Resolution set in the Game View.

World Space however you get full control over as it behaves as a regular game object.



Canvas Scalar

Canvas Scalar determine how to UI elements are scaled you can do it in 3 ways, using pixels, by scale to a specific resolution or by using physical dimensions. Either one will let you scale the size of Images, Text and buttons that the Canvas holds.



Canvas Scaler
UI Scale Mode: Constant Pixel Size
Scale Factor: 1
Reference Pixels Per: 100

Canvas Scaler
UI Scale Mode: Scale With Screen Size
Reference Resolution X: 800 Y: 600
Screen Match Mode: Match Width Or Height
Match: 0
Width Height
Reference Pixels Per: 100

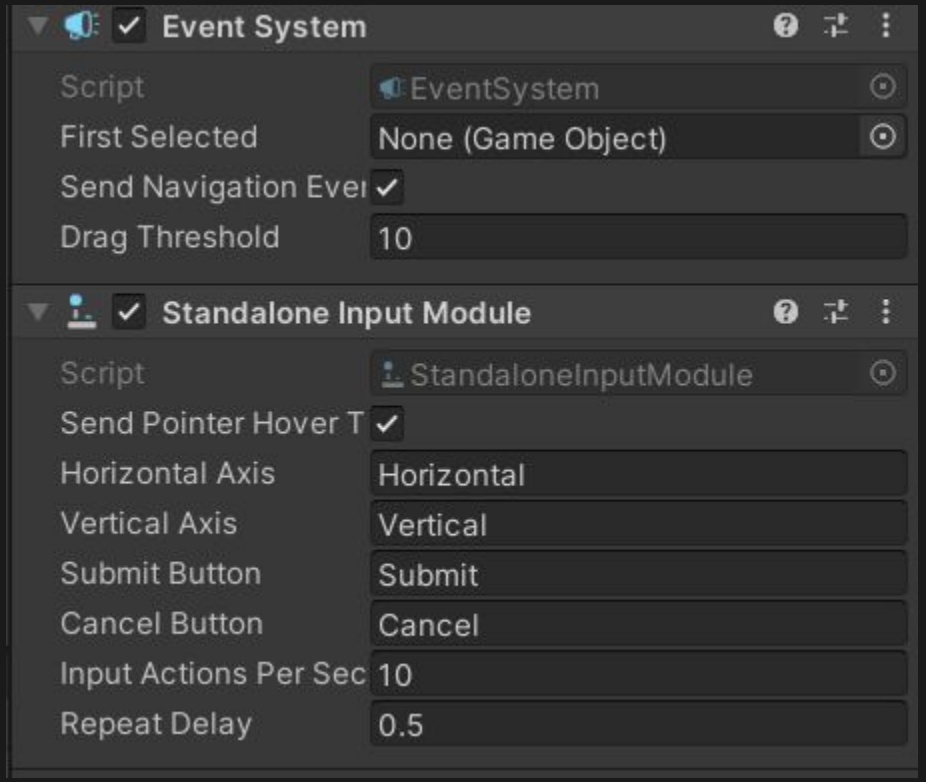
Canvas Scaler
UI Scale Mode: Constant Physical Size
Physical Unit: Points
Fallback Screen DPI: 96
Default Sprite DPI: 96
Reference Pixels Per: 100

Sebastian Grygorczuk - STEM Institute at CCNY

Event System

When you create a Canvas automatically the Event System Game Object will also be created.

The Event System Game Object allows the Canvas to take in user Input and perform actions based on them.



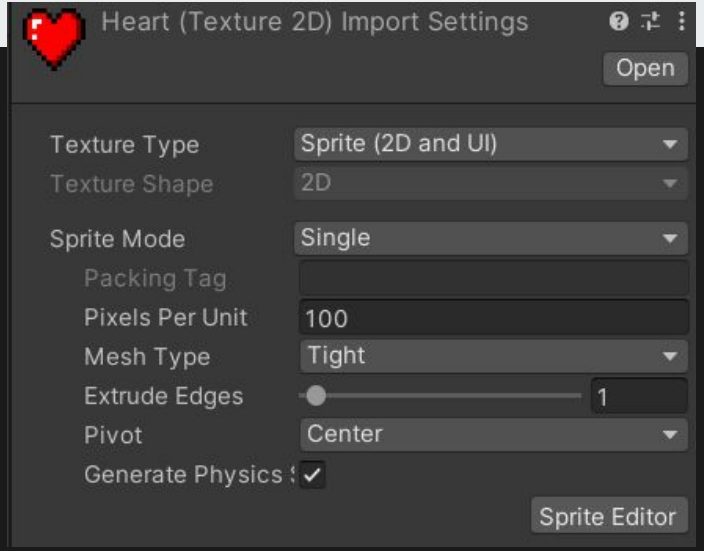
Images

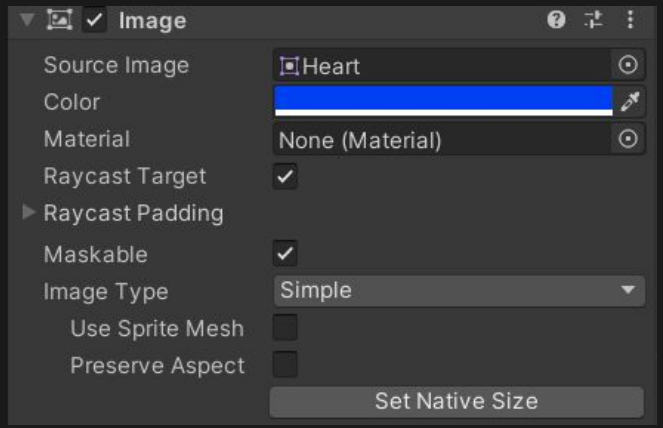
When placing a image down into the UI you first have to make sure that its Import Settings are set to Sprite (2D and UI) otherwise it will be imported as a Texture and won't show up on the Source Image List.

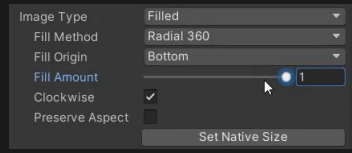
The Image shares a lot of common aspect as the Sprite Renderer.

And has one more Image Type, in addition to sliced, simple, tiled now we also have field which can control how much of the image is visible.

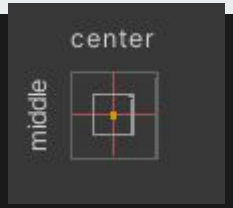
This is very useful for showing player health, stamina or other stats.

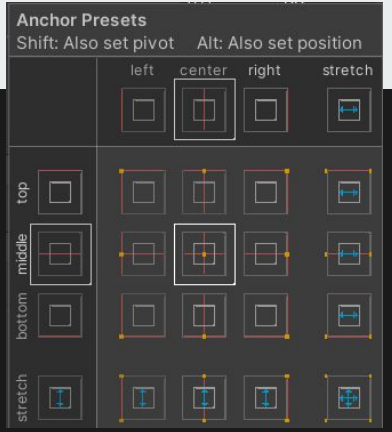






Anchors

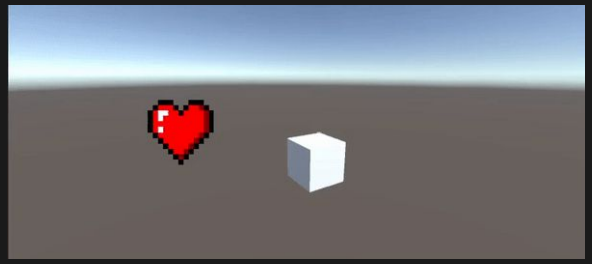


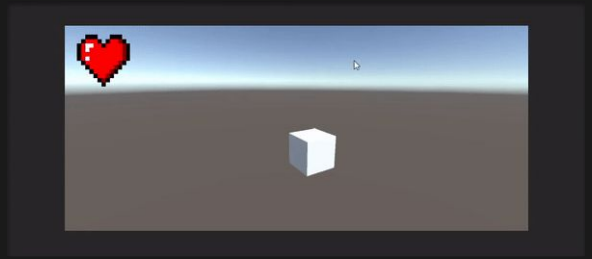


Inside the Rect Transform for the Image you may have found this little Menu Item.

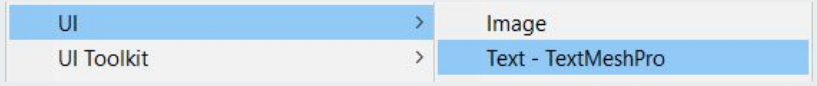
This is the Anchor it determines how the UI element should behave when. All UI elements come in with the Center behavior enabled meaning that the item will more or less stay in the same position of the screen regardless of the size of the screen.

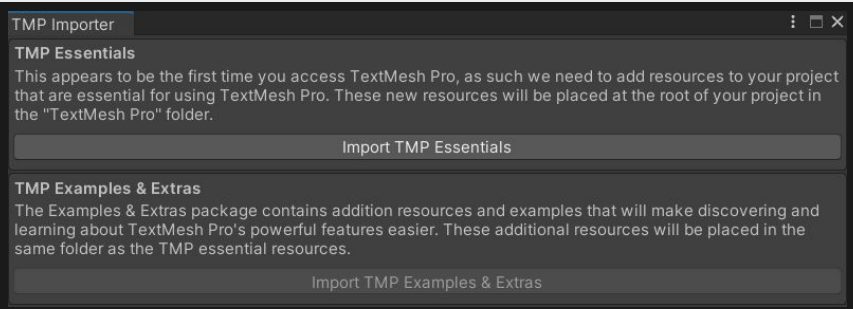
As you can see that's not very useful, it's best to attach an anchor to a corner and then the item will always move in reference to that corner.

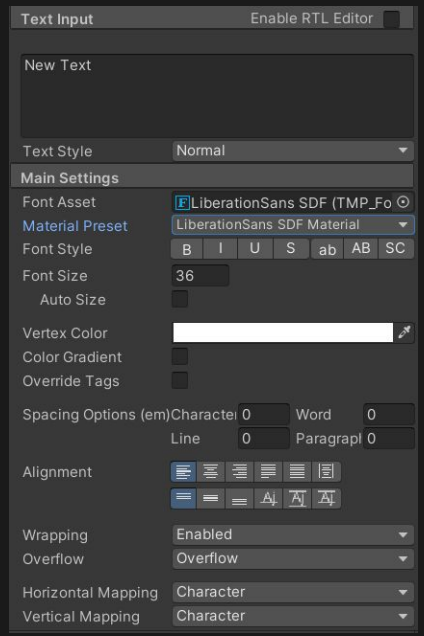




Text







Text is an important part of the game to let the player know what to do.

When you first create a Text you will get a Popup to install TextMesh Pro Essential, click yes and that will bring in basic fonts to be drawn and will create a folder with the font game assets.

Once all of that is in you will be met with the Text Input and you will have the big text box where you can type in what will show up with many settings that you should recognize from word processing softwares.

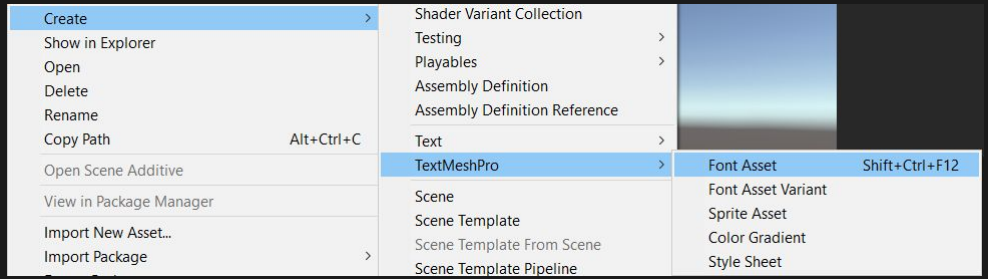
Sebastian Grygorczuk - STEM Institute at CCNY

Fonts

https://www.1001freefonts.com/

When you install the Mesh Pro Text you will only be met with one Font type, if you want your text to look different you will have to download it from a website like 1001 free fonts.

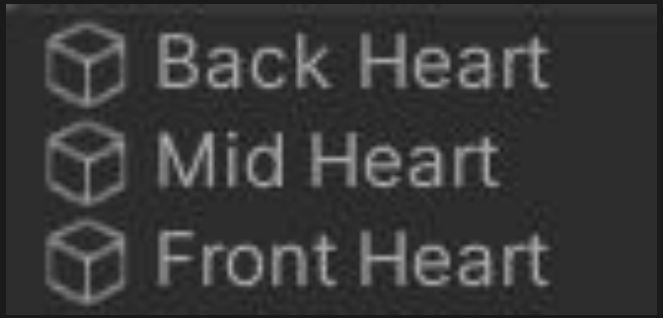
Once you have it download and imported the font you want you then have to process it into a Pro Mesh Text before you can apply it to a Text Game Object.



Layering Object on Canvas

Unlike 3D where the Z axis or 2D where Layer Order controls what is draw closer to the camera the order of the UI elements inside the Hierarchy determines the draw order.

Game Object that are higher in the child list will be draw in the back while the Game Object that are lower on the child list will be drawn in the font.

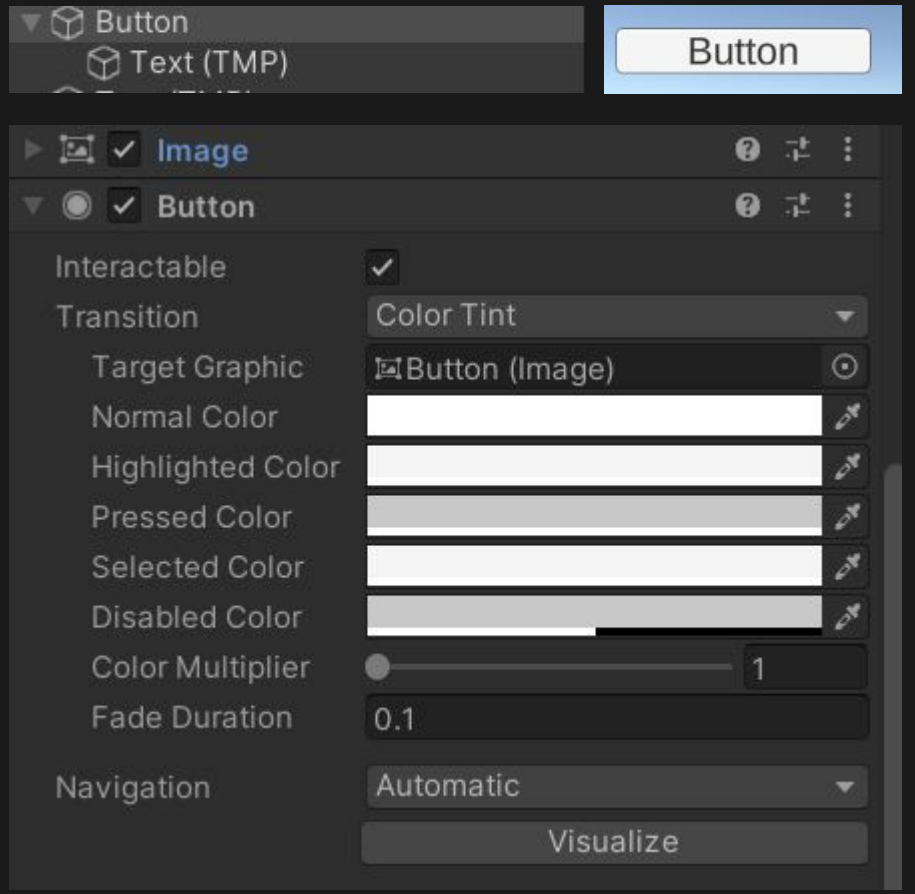


Buttons

Creating a button will also create a Text Game Object that will display the text on the button.

The button also comes with an Image Component where you can replace the image of the button.

Finally the button component gives you the ability to click it how the image should behave after interacting with it.



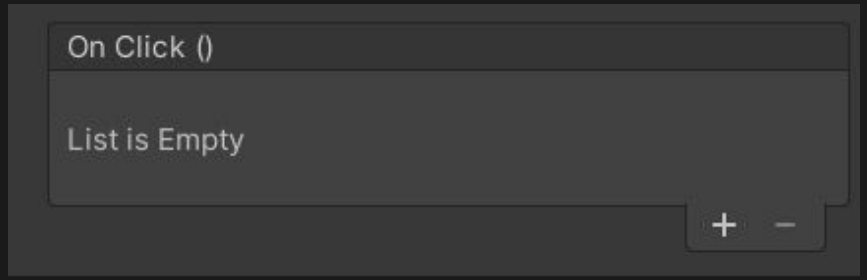
Connect Button Action To Text

To be able to connect a script to a button you will use this OnClick section bellow the color selection.

First you will need your script, we're going to make this button show an number of times you've click on something.

So here we have a script will require that you add a new using TMPro, so that we can connect to the Text which is TextMesh Pro GUI class.

Then we just update the button counter each time it's clicked and set the text equal to that new updated value.




```
//You have to be using TMPro to connect
//To the GUI elements
using TMPro;
```

```
//Counts how many times the button has been pressed
private int _buttonCounter = 0;
//Connect to the Text On Screen that will update
public TextMeshProUGUI textMesh;

//Increments the button Counter and update the text shown in the
//Text box
0 references
public void UpdateText()
{
    _buttonCounter++;
    textMesh.text = "Score: " + _buttonCounter;
}
```

Sebastian Grygorczuk - STEM Institute at CCNY

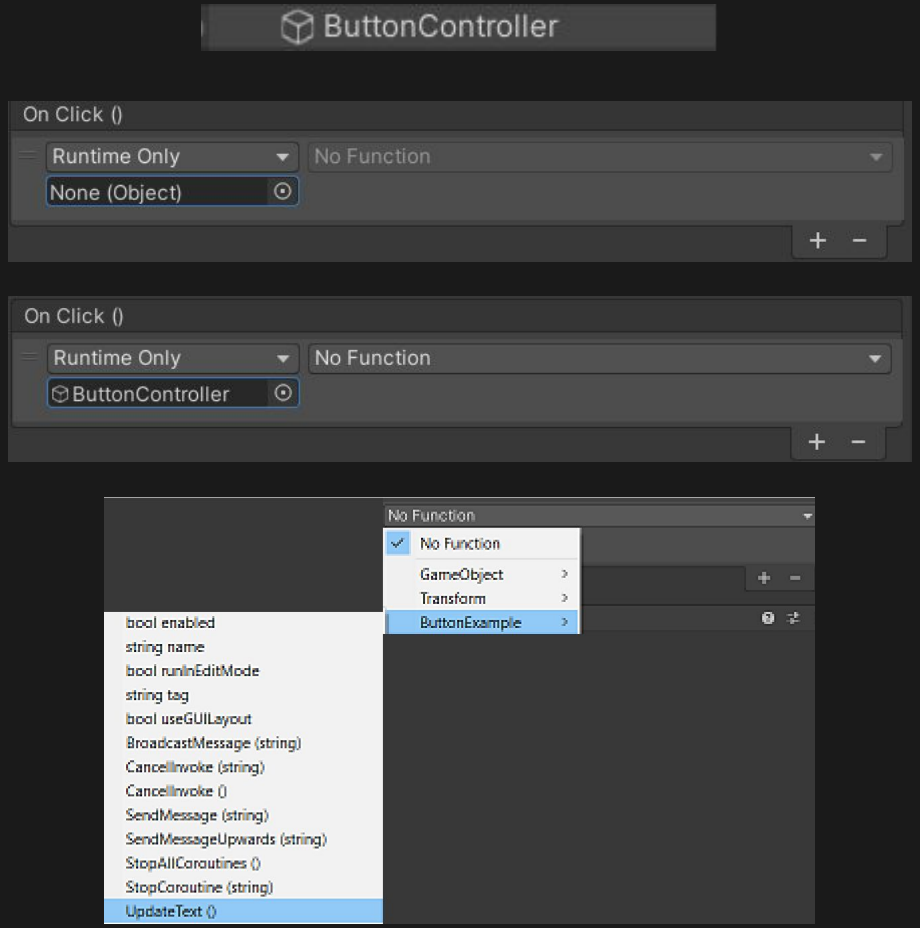
Connect Button Action To Text

To connect the script you will have to have a game object in the heichy that has the script connected to it, in this case Button Controller.

Then you can click on the plus button which will unlock a space to add an object into.

You will drag the game object into that space.

Finally you will be able to select the script component and pull the Update Text Method we created.



Sebastian Grygorczuk - STEM Institute at CCNY

Using Script to Move Between Scenes

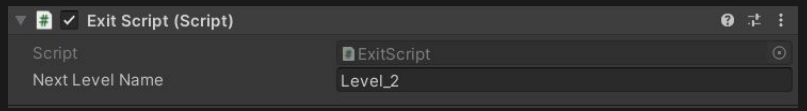
You can move between scene you'll have to set up a using UnityEngine.SceneManagement;

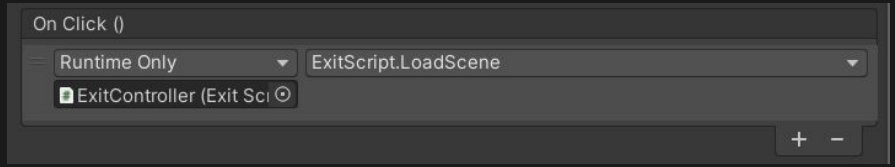
Once you have that you'll be able to call a SceneManager class and using the LoadScene you can pass in a string with the name of the level you want to go to.

Following the example of the button we can set up movement between scene in the same fashion.

```
using UnityEngine.SceneManagement;
```

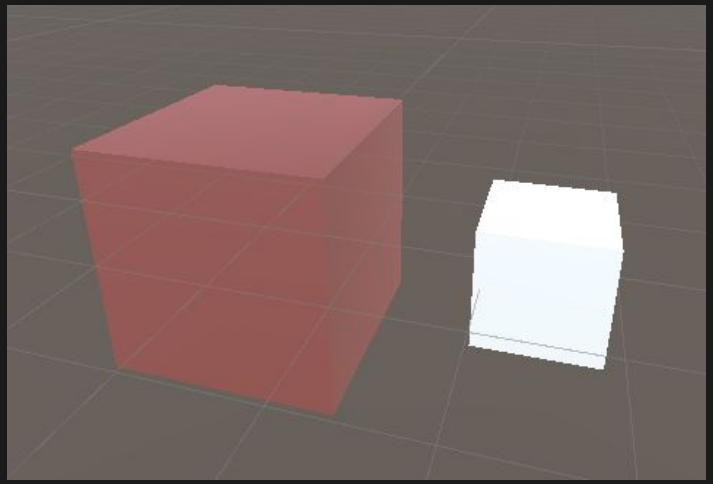
```
//Used by the button
1 reference
public void LoadScene()
{
    SceneManager.LoadScene(nextLevelName);
}
```





Using Script to Move Between Scenes

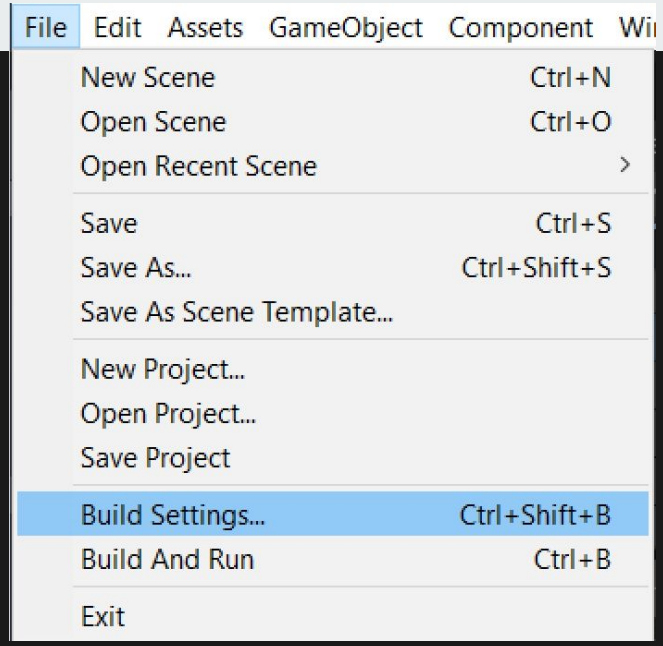
Unlike the button we can also connect this function to a Trigger and have a character walk into a collider and use it to move between scenes.

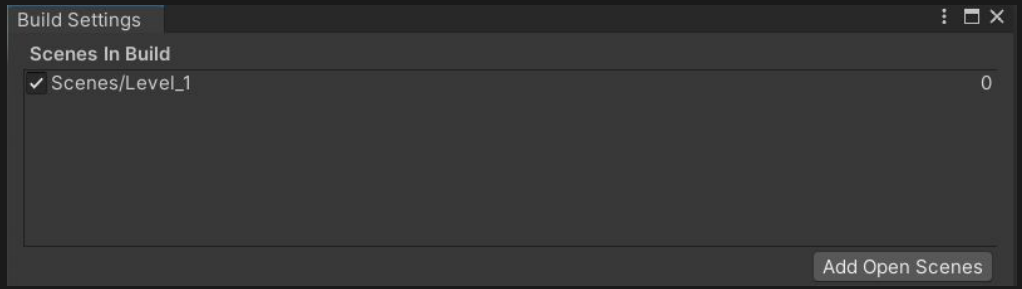


```
//Used by a Trigger Collider
Unity Message | 0 references
private void OnTriggerEnter(Collider other)
{
    if (other.CompareTag("Player"))
    {
        LoadScene();
    }
}
```

Build Settings

For you to be able to move between scene the scene have to be added in the Scene in Build in the Build Setting inside the File Drop down Menu





Using Script Save Data Between Scenes

```
private static Data _instance;
```

To save data between level you need to call a copy of the class inside itself. So here I have a script called Data and I make a copy of it inside the start function.

I check to see if one exists since at the start of a scene since we don't want multiples of them exiting each time we enter a new level.

We destroy the old one, and set this new one as the used copy.

Then we set it to DontDestoryOnLoad allowing us to transfer data between class.es

```
Unity Message | 0 references
private void Start()
{
    //Checks if there already exits
    if (_instance != null)
    {
        Destroy(gameObject);
        return;
    }

    _instance = this;
    DontDestroyOnLoad(gameObject);
```