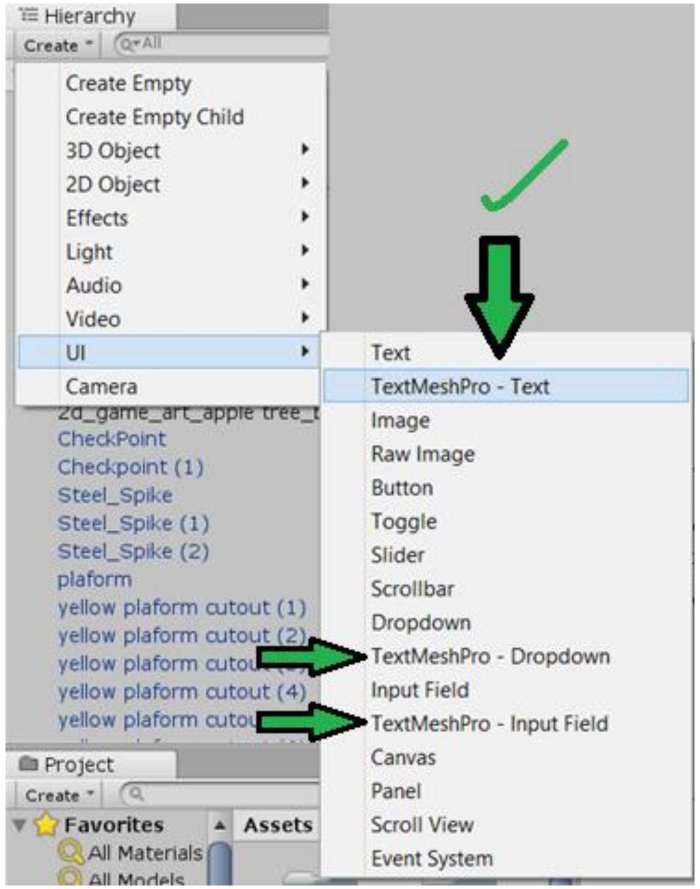
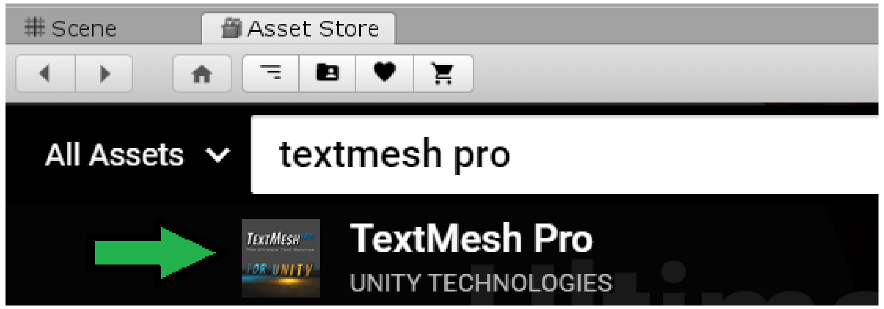
Character Dialogue

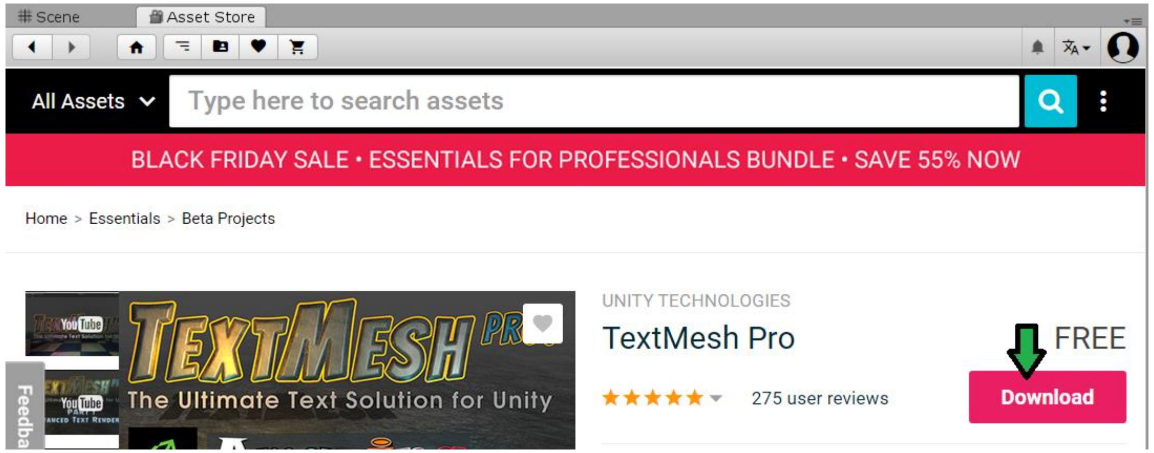
9

Exercise #39 – Confirm TextMesh Pro is Installed

- If these TextMesh Pro options do not exist, then install TextMesh Pro from Unity's Asset Store

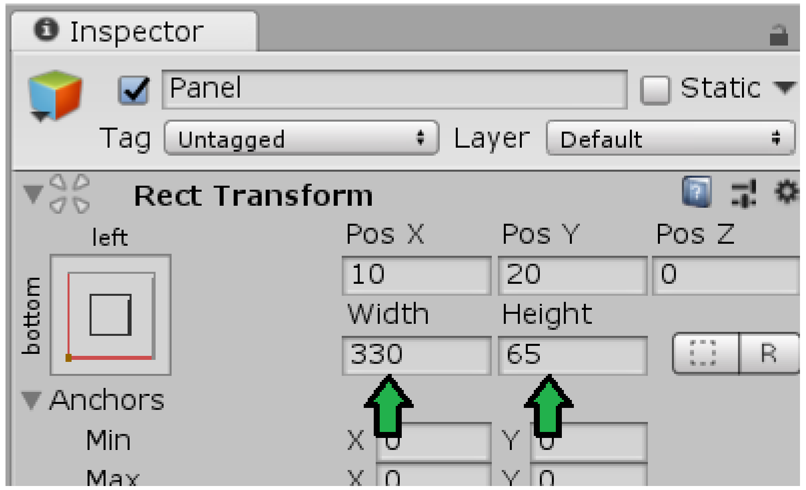





Scene
Asset Store
All Assets
Type here to search assets
BLACK FRIDAY SALE • ESSENTIALS FOR PROFESSIONALS BUNDLE • SAVE 55% NOW
Home > Essentials > Beta Projects
TEXTMESH PRO
The Ultimate Text Solution for Unity
UNITY TECHNOLOGIES
TextMesh Pro
275 user reviews
FREE
Download

Exercise #40 – Create Dialog Box

- You will need a panel to serve as the dialogue box, so the text appears clearly on screen
- Go to Create → UI → and select Panel
- Set the color of the Panel to dark grey and set the alpha to 245
- Set the width and height of the Panel to 330 and 65, respectively



Inspector

Panel — Static

Tag: Untagged — Layer: Default

Rect Transform

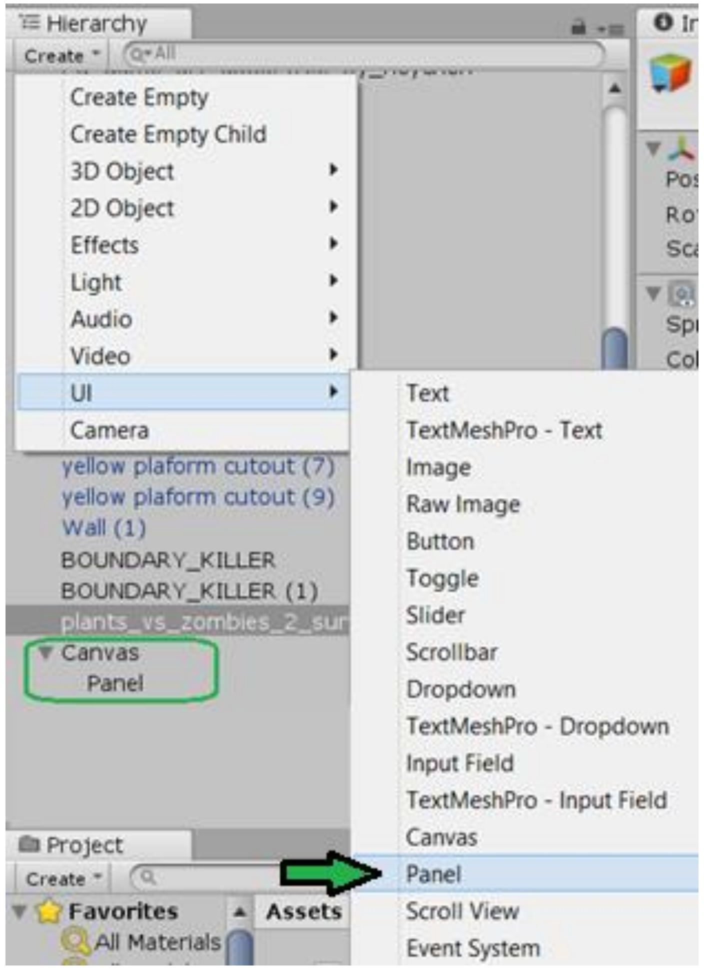
left / bottom

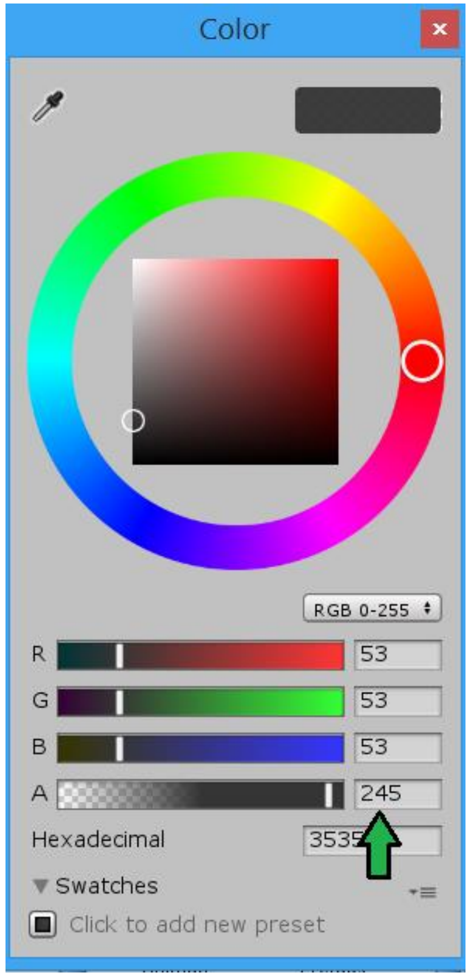
Pos X	Pos Y	Pos Z
10	20	0
Width	Height	
330	65	

Anchors

Min — X 0 — Y 0

Max — X 0 — Y 0





Color

RGB 0-255

R 53

G 53

B 53

A 245

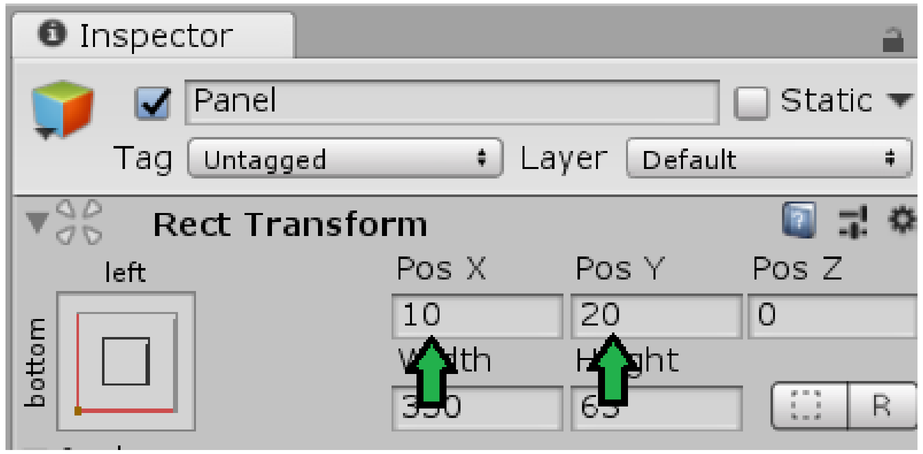
Hexadecimal 3535

Swatches

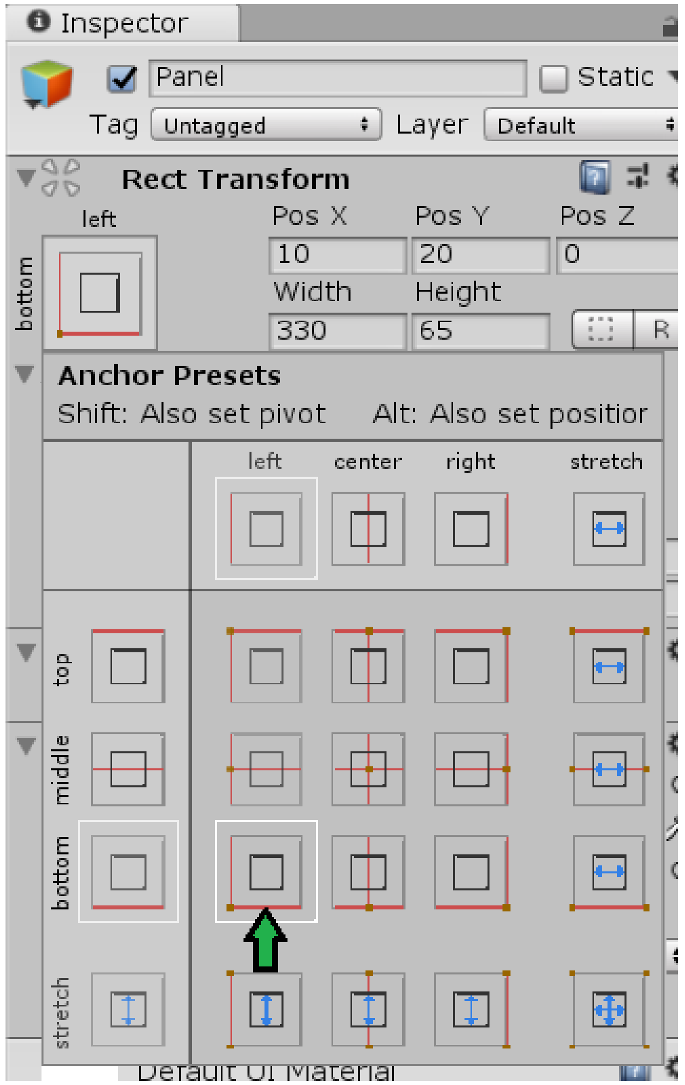
Click to add new preset

Exercise #40 – Create Dialog Box (Cont.)

- Set the anchor, pivot, and position to bottom left (hint: hold shift and alt to adjust the pivot and position)
- Adjust the position of x to 10 and y to 20 so that the panel is not stuck to the edges of the screen
- **These values are tuned to this example, you need to experiment with the values depending on your project and design**



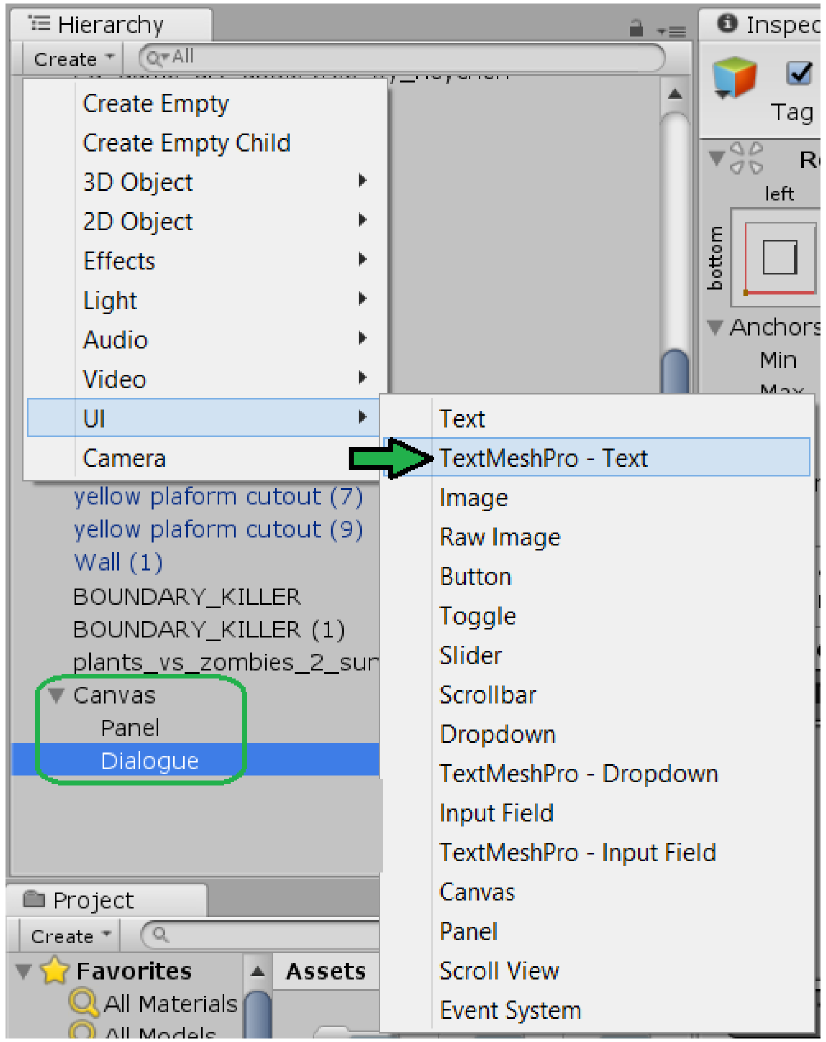
Panel
Tag Untagged
Layer Default
left
bottom
Pos X 10
Pos Y 20
Pos Z 0
Width 330
Height 65



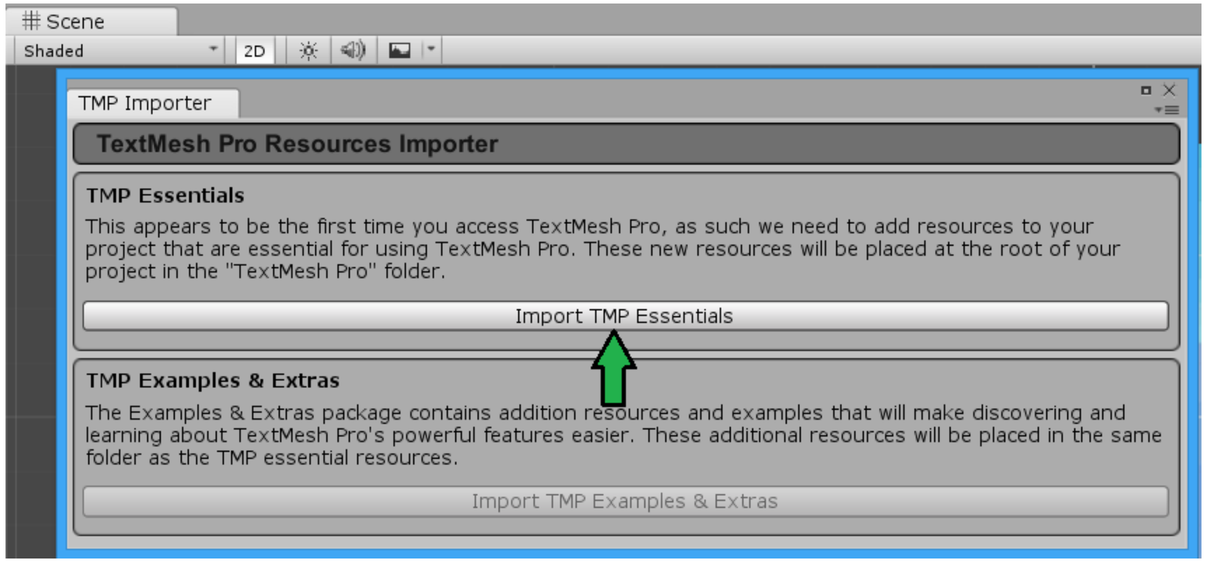
Panel
Tag Untagged
Layer Default
left
bottom
Pos X 10
Pos Y 20
Pos Z 0
Width 330
Height 65

Shift: Also set pivot Alt: Also set position

Exercise #41 – Create Dialog Text



- Go to Create → UI → and select TextMeshPro – Text
- You will be asked to import the TMP Essentials, <u>import them</u>
- Your Assets folder have a new folder named “TextMesh Pro”
- And your text should be at the center of the canvas/screen



TextMesh Pro Resources Importer

TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

Import TMP Essentials

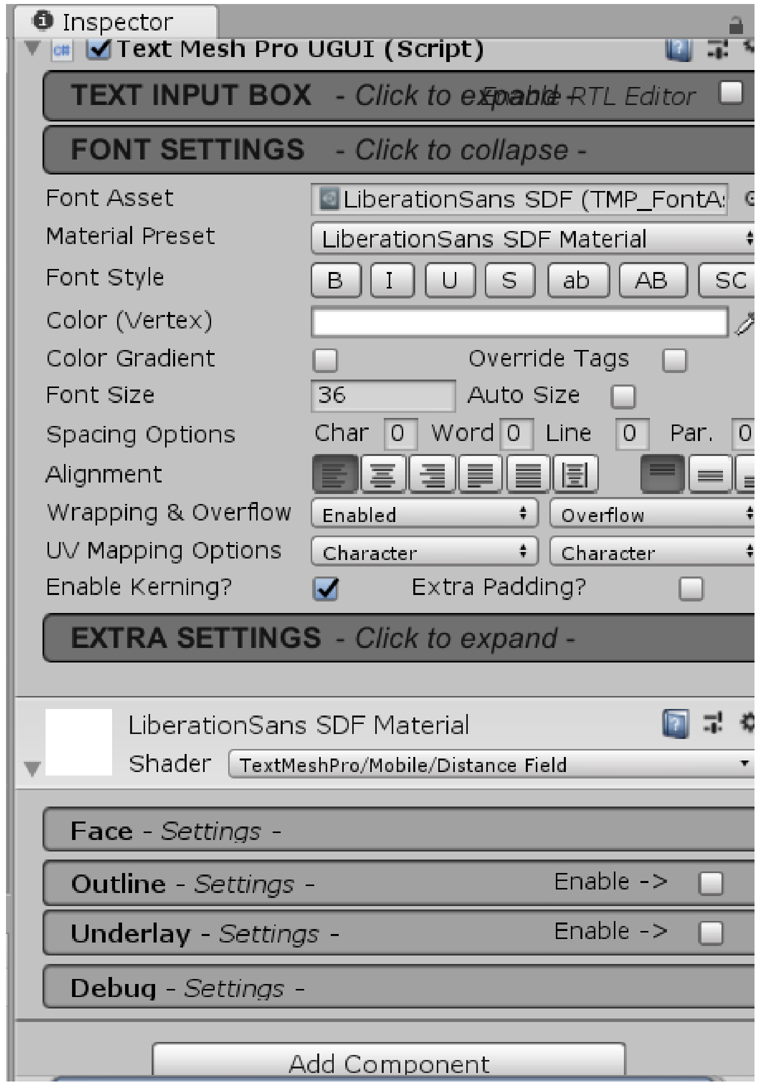
TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

Import TMP Examples & Extras

Exercise #41 – Create Dialog Text (Cont.)

- TextMeshPro – Text has a lot of settings to customize the displayed text
  - The standard UI settings such as anchors, position, width, and height
  - TEXT INPUT BOX: the text that is written in this field you will be the one that is displayed on screen
  - FONT SETTINGS: changing the values in this section affects the look of the displayed text such as its size, color, style, alignment, etc.
  - You can find other settings that help in designing your own style of text, such as Underlay which applies a shadow to the displayed text



Inspector

Text Mesh Pro UGUI (Script)

TEXT INPUT BOX - Click to expand - Enable RTL Editor

FONT SETTINGS - Click to collapse -

Font Asset	LiberationSans SDF (TMP_FontA
Material Preset	LiberationSans SDF Material
Font Style	B I U S ab AB SC
Color (Vertex)	
Color Gradient	☐ Override Tags ☐
Font Size	36 Auto Size ☐
Spacing Options	Char 0 Word 0 Line 0 Par. 0
Alignment	
Wrapping & Overflow	Enabled / Overflow
UV Mapping Options	Character / Character
Enable Kerning?	✓ Extra Padding? ☐

EXTRA SETTINGS - Click to expand -

LiberationSans SDF Material

Shader TextMeshPro/Mobile/Distance Field

Face - Settings -

Outline - Settings - Enable -> ☐

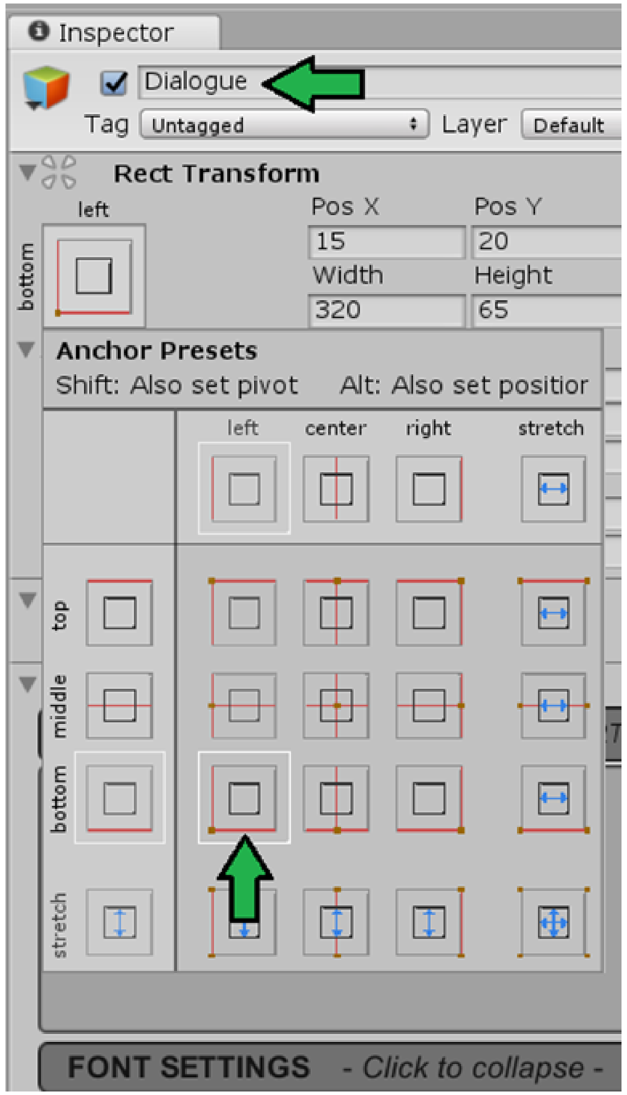
Underlay - Settings - Enable -> ☐

Debug - Settings -

Add Component

Exercise #41 – Create Dialog Text (Cont.)

- You will need your text to be on top of the panel (dialogue box) and centered
- Rename your TextMeshPro – Text gameObject to Dialogue
- Set the anchor, pivot, and position to bottom left (hint: hold shift and alt to adjust the pivot and position)
- Set the width and height of the text to 320 and 65, respectively
- Adjust the position of x to 15 and y to 20 so that the text is not stuck to the edges of the box
- Remove the text that is in the TEXT INPUT BOX setting



Inspector

Dialogue

Tag Untagged Layer Default

Rect Transform

left

bottom

Pos X 15

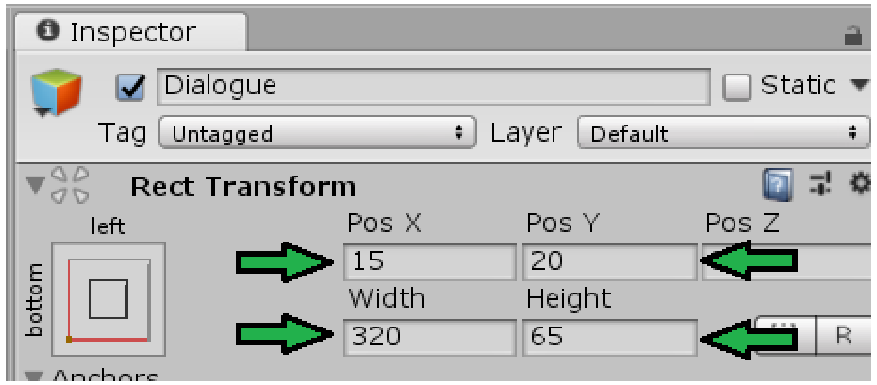
Pos Y 20

Width 320

Height 65

Anchor Presets

Shift: Also set pivot Alt: Also set position



Inspector

Dialogue Static

Tag Untagged Layer Default

Rect Transform

left

bottom

Pos X 15

Pos Y 20

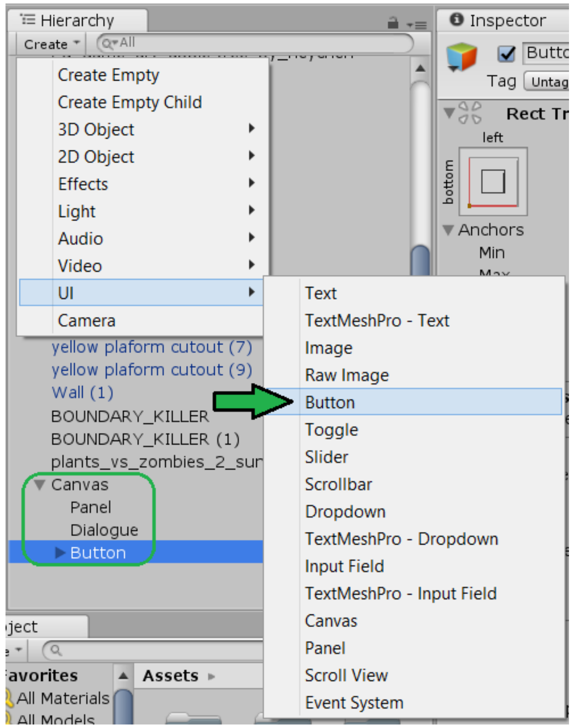
Pos Z

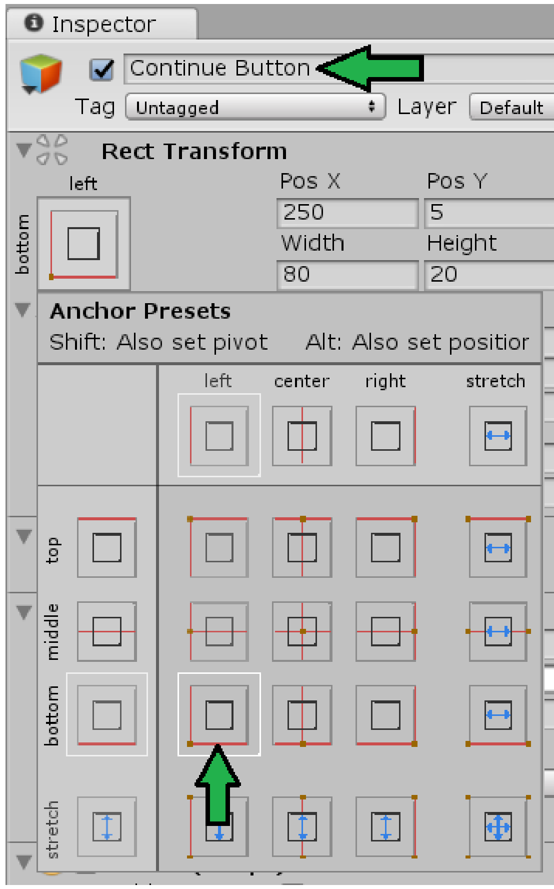
Width 320

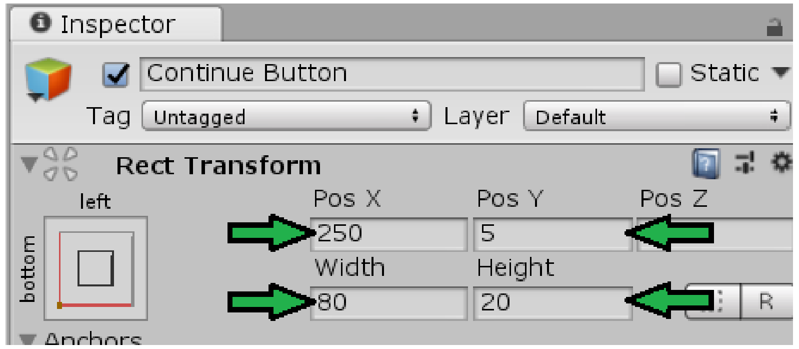
Height 65

Exercise #42 – Create Continue Button

- You need to have the ability to progress through the dialogue, as there will be more than one sentence in a dialogue
- Create → UI → Button
- Rename your Button gameobject to Continue Button
- Set the anchor, pivot, and position to bottom left (hint: hold shift and alt to adjust the pivot and position)
- Set the Width and Height of the button to 80 and 20, respectively
- Adjust the position of x to 250 and y to 5 so that the button does not overlap the dialogue text

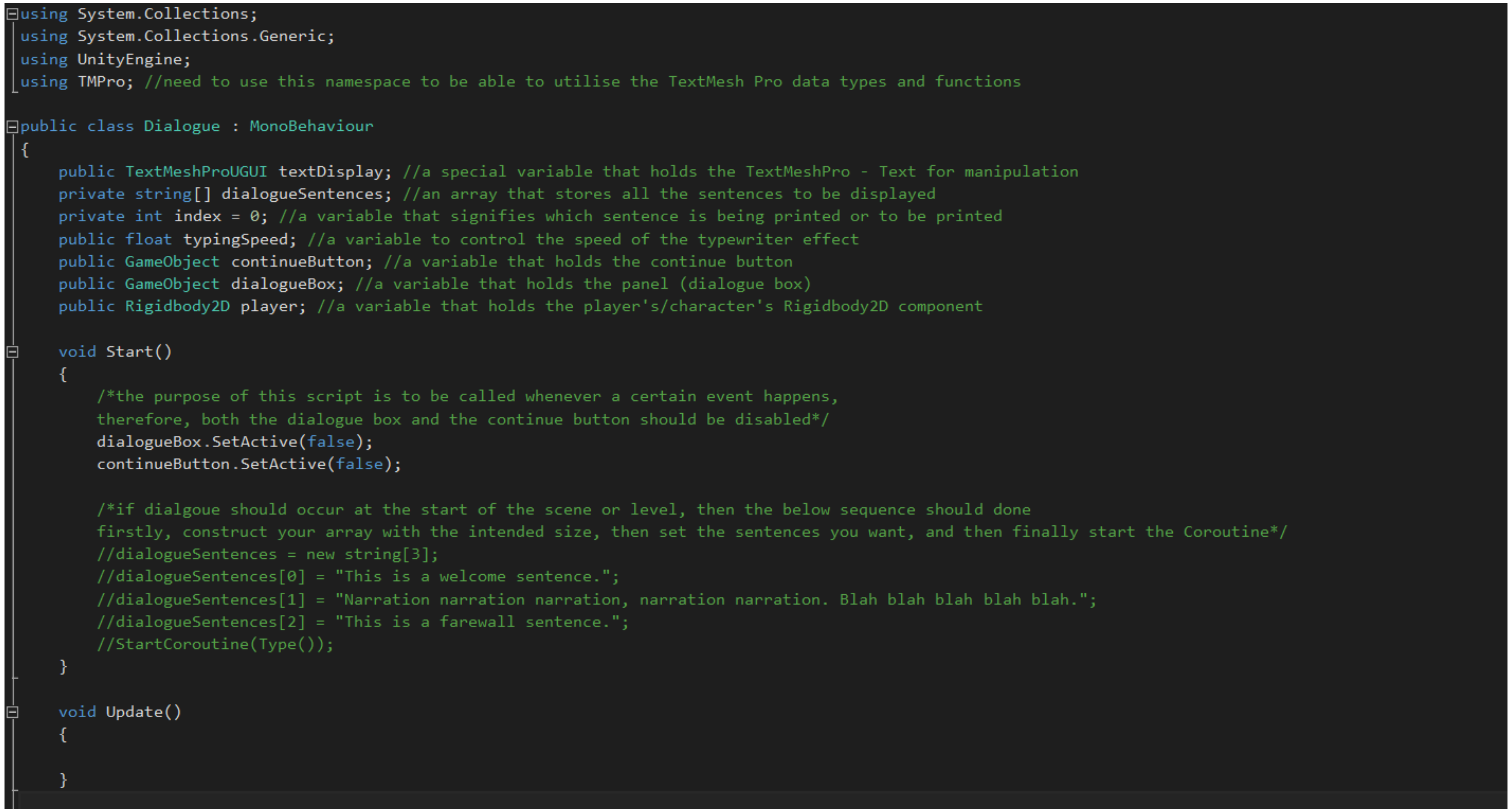






Exercise #43 – Create Dialogue Script

- Create a new script named Dialogue and write the following



```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;
using TMPro; //need to use this namespace to be able to utilise the TextMesh Pro data types and functions

public class Dialogue : MonoBehaviour
{
    public TextMeshProUGUI textDisplay; //a special variable that holds the TextMeshPro - Text for manipulation
    private string[] dialogueSentences; //an array that stores all the sentences to be displayed
    private int index = 0; //a variable that signifies which sentence is being printed or to be printed
    public float typingSpeed; //a variable to control the speed of the typewriter effect
    public GameObject continueButton; //a variable that holds the continue button
    public GameObject dialogueBox; //a variable that holds the panel (dialogue box)
    public Rigidbody2D player; //a variable that holds the player's/character's Rigidbody2D component

    void Start()
    {
        /*the purpose of this script is to be called whenever a certain event happens,
        therefore, both the dialogue box and the continue button should be disabled*/
        dialogueBox.SetActive(false);
        continueButton.SetActive(false);

        /*if dialgoue should occur at the start of the scene or level, then the below sequence should done
        firstly, construct your array with the intended size, then set the sentences you want, and then finally start the Coroutine*/
        //dialogueSentences = new string[3];
        //dialogueSentences[0] = "This is a welcome sentence.";
        //dialogueSentences[1] = "Narration narration narration, narration narration. Blah blah blah blah blah.";
        //dialogueSentences[2] = "This is a farewall sentence.";
        //StartCoroutine(Type());
    }

    void Update()
    {

    }
```

Exercise #43 – Create Dialogue Script (Cont.)

- For our next code, we will need to a Coroutine function. If you are unfamiliar with what that is, you are advised to read more about them in the manual
    - *https://docs.unity3d.com/Manual/Coroutines.html*

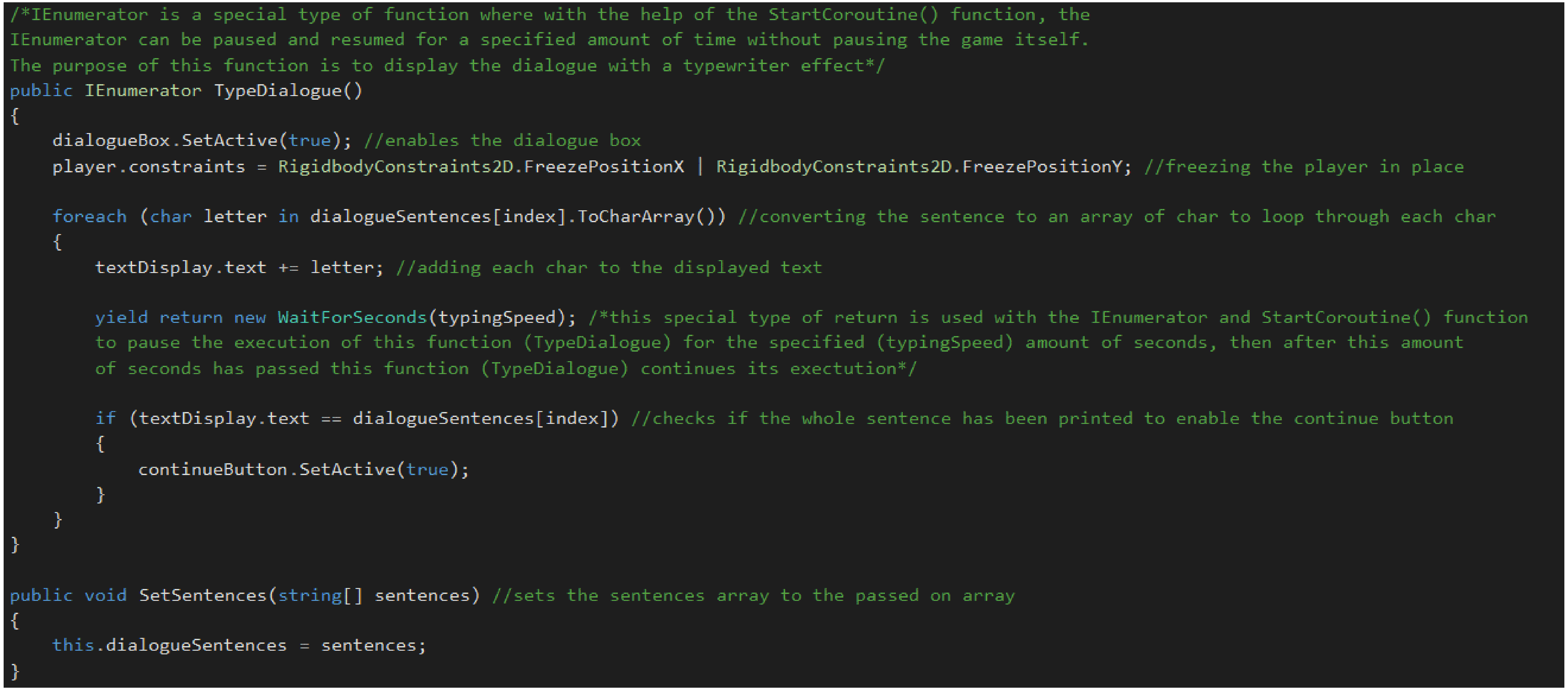
- Coroutine has the ability to interrupt normal execution, do something, then return control to Unity to continue where it left off on the following frame. It’s used with a .NET type of IEnumerator which can be used to fragment large collection or files
    - Therefore Coroutine is a way to spread an effect over a period of time, such as a cutscene, dialog scene or special effects, like fading animation. Those things do not always work well using the Update function because they’re *not* supposed to be called every frame!

Exercise #43 – Create Dialogue Script (Cont.)

- To run Coroutine for creating a dialog, we need to use a function of IEnumerator type

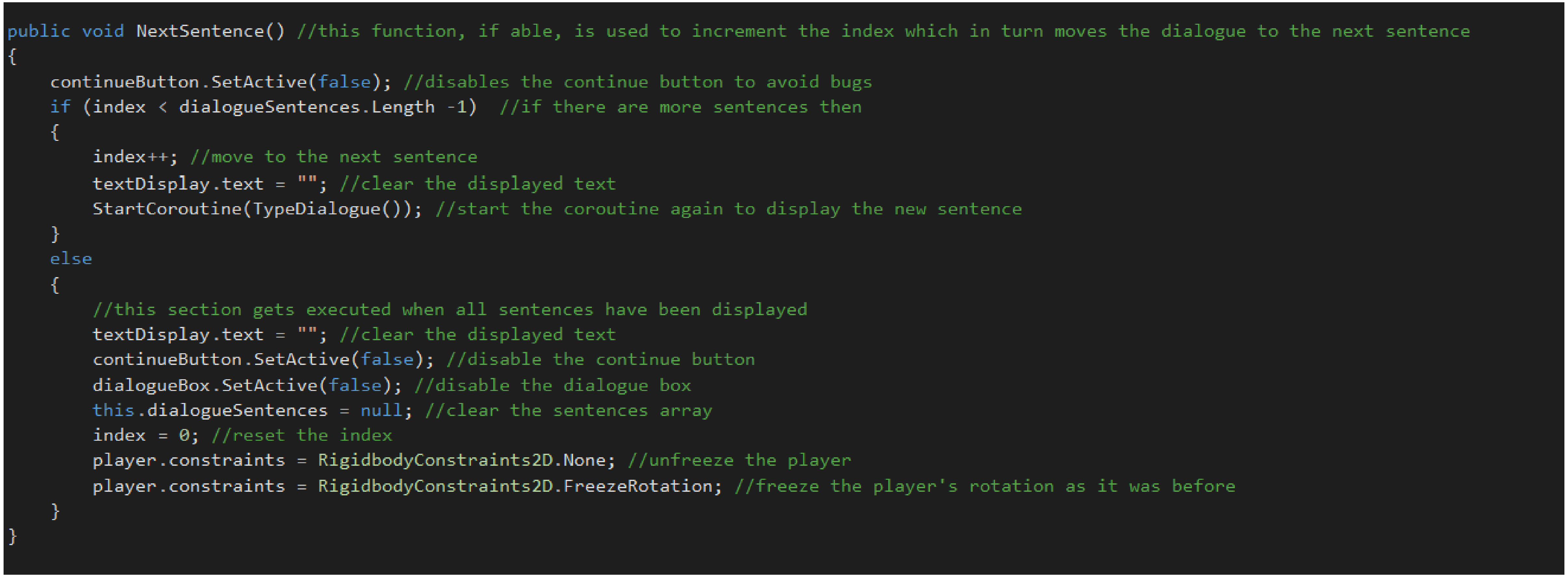
- To control the speed with which the dialog is appearing on the screen, we will need to add the word *yield* before return*.* So we will write something like:
    - *yield return new WaitforSeconds(time delay value (e.g. 0.5 seconds))*

Exercise #43 – Create Dialogue Script (Cont.)



```
/*IEnumerator is a special type of function where with the help of the StartCoroutine() function, the
IEnumerator can be paused and resumed for a specified amount of time without pausing the game itself.
The purpose of this function is to display the dialogue with a typewriter effect*/
public IEnumerator TypeDialogue()
{
    dialogueBox.SetActive(true); //enables the dialogue box
    player.constraints = RigidbodyConstraints2D.FreezePositionX | RigidbodyConstraints2D.FreezePositionY; //freezing the player in place

    foreach (char letter in dialogueSentences[index].ToCharArray()) //converting the sentence to an array of char to loop through each char
    {
        textDisplay.text += letter; //adding each char to the displayed text

        yield return new WaitForSeconds(typingSpeed); /*this special type of return is used with the IEnumerator and StartCoroutine() function
        to pause the execution of this function (TypeDialogue) for the specified (typingSpeed) amount of seconds, then after this amount
        of seconds has passed this function (TypeDialogue) continues its exectution*/

        if (textDisplay.text == dialogueSentences[index]) //checks if the whole sentence has been printed to enable the continue button
        {
            continueButton.SetActive(true);
        }
    }
}

public void SetSentences(string[] sentences) //sets the sentences array to the passed on array
{
    this.dialogueSentences = sentences;
}
```

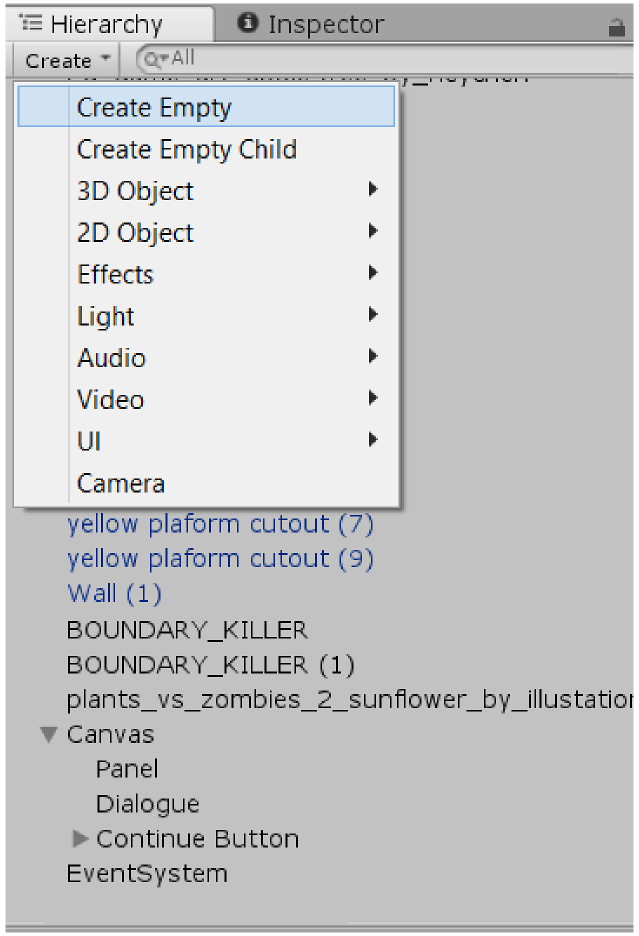
Exercise #43 – Create Dialogue Script (Cont.)

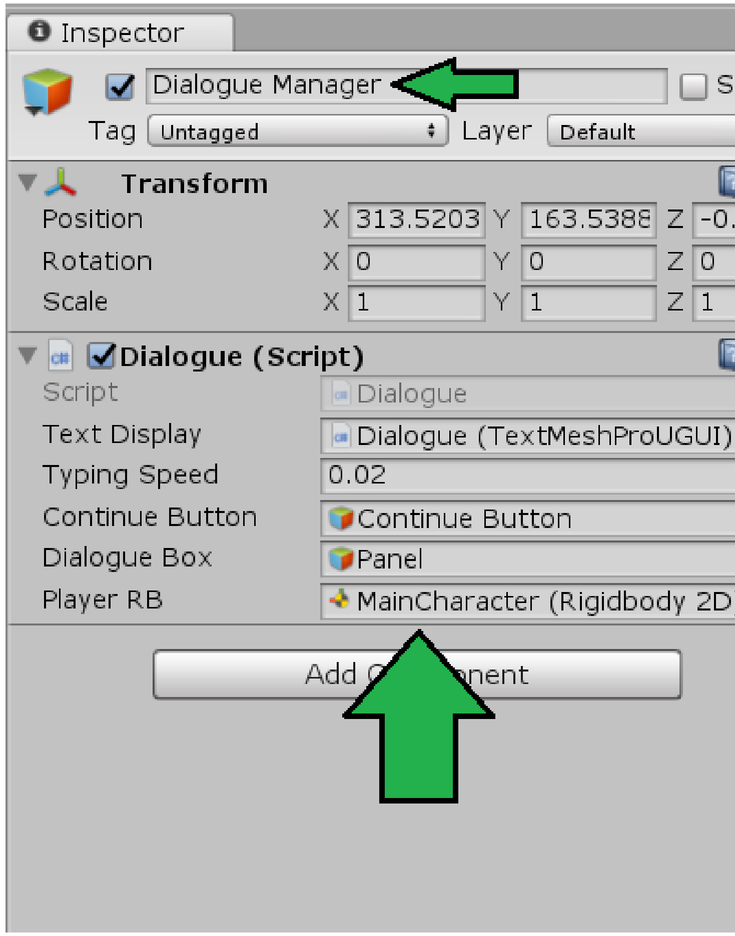


```csharp
public void NextSentence() //this function, if able, is used to increment the index which in turn moves the dialogue to the next sentence
{
    continueButton.SetActive(false); //disables the continue button to avoid bugs
    if (index < dialogueSentences.Length -1)  //if there are more sentences then
    {
        index++; //move to the next sentence
        textDisplay.text = ""; //clear the displayed text
        StartCoroutine(TypeDialogue()); //start the coroutine again to display the new sentence
    }
    else
    {
        //this section gets executed when all sentences have been displayed
        textDisplay.text = ""; //clear the displayed text
        continueButton.SetActive(false); //disable the continue button
        dialogueBox.SetActive(false); //disable the dialogue box
        this.dialogueSentences = null; //clear the sentences array
        index = 0; //reset the index
        player.constraints = RigidbodyConstraints2D.None; //unfreeze the player
        player.constraints = RigidbodyConstraints2D.FreezeRotation; //freeze the player's rotation as it was before
    }
}
```

Exercise #44 – Create Dialogue Manager

- You will need to attach the Dialogue Script to a gameobject
- Create → Create Empty
- Rename your empty gameobject to <u>Dialogue Manger</u>
- Attach the Dialogue Script
- Set the variables of the script
  - Drag and drop the dialogue
  - Drag and drop the continue button
  - Drag and drop the dialogue box (panel)
  - Drag and drop the Rigidbody2D of the character/player
  - Set typing speed to 0.02





Dialogue Manager
Tag Untagged
Layer Default

Transform

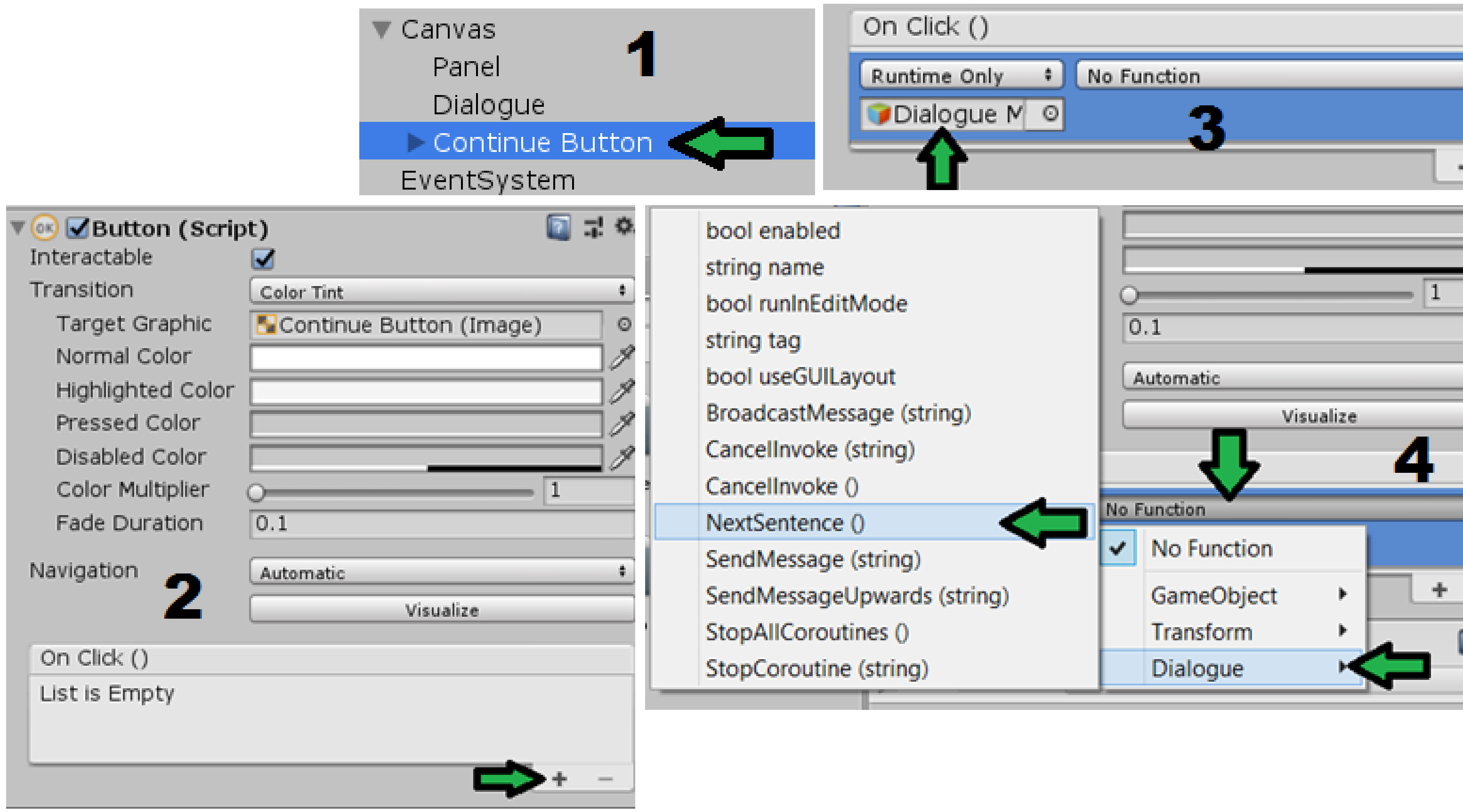
Position	X	313.5203	Y	163.5388	Z	-0.
Rotation	X	0	Y	0	Z	0
Scale	X	1	Y	1	Z	1

Dialogue (Script)

Script	Dialogue
Text Display	Dialogue (TextMeshProUGUI)
Typing Speed	0.02
Continue Button	Continue Button
Dialogue Box	Panel
Player RB	MainCharacter (Rigidbody 2D

Exercise #45 – Ready the Continue Button

- You will need to make the continue button functional
- Select the Continue Button
- Press the + sign to add an OnClick() event
- Drag and drop the Dialogue Manager gameobject
- Select the function NextSentence() from the dropdown menu


Canvas
Panel
Dialogue
Continue Button
EventSystem

Button (Script)
Interactable ✓
Transition: Color Tint
Target Graphic: Continue Button (Image)
Normal Color
Highlighted Color
Pressed Color
Disabled Color
Color Multiplier: 1
Fade Duration: 0.1
Navigation: Automatic
Visualize
On Click ()
List is Empty

On Click ()
Runtime Only | No Function
Dialogue M

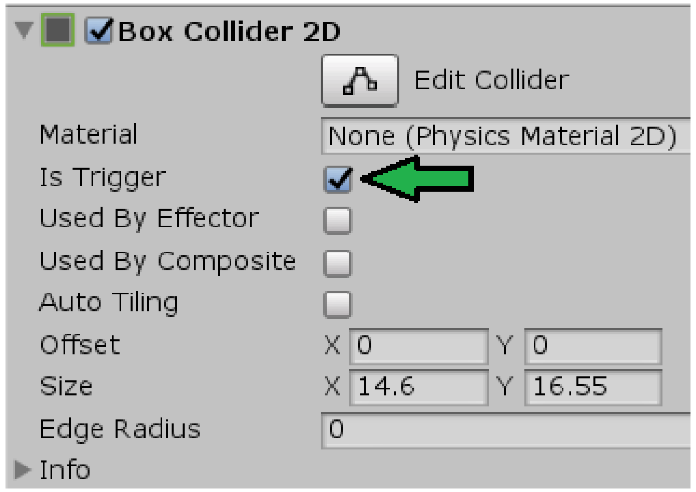
Exercise #46 – Create Character in Dialogue

- The simplest and easiest way to trigger dialogue is using the OnTriggerEnter2D() function

- Choose a sprite/character that you want your character to talk to

- For our example we are going to use a flower

- Add a collider2D and set its IsTrigger to true



Box Collider 2D [✓]

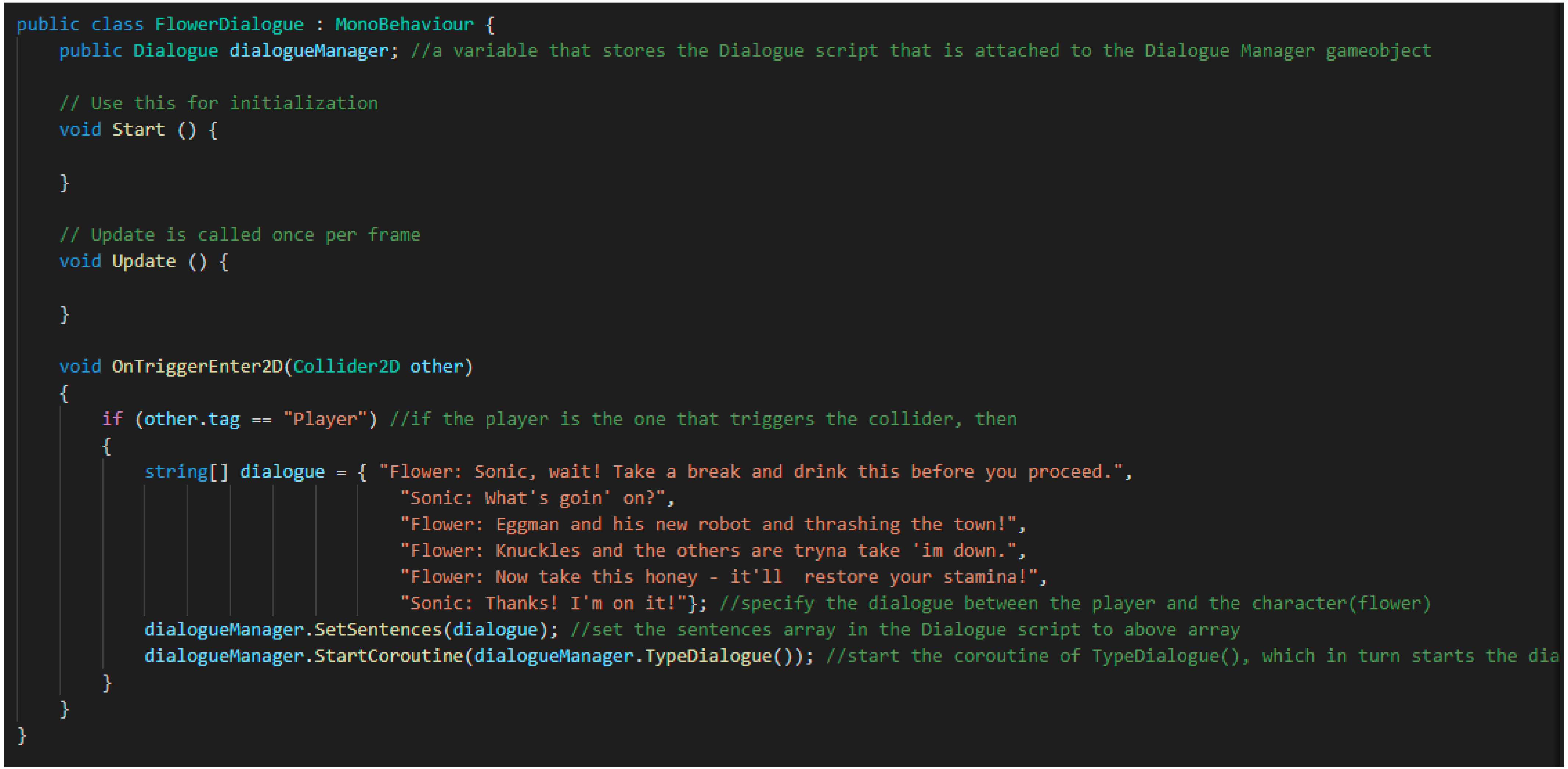
Edit Collider

Material	None (Physics Material 2D)
Is Trigger	✓
Used By Effector	☐
Used By Composite	☐
Auto Tiling	☐
Offset	X 0 Y 0
Size	X 14.6 Y 16.55
Edge Radius	0

Info

Exercise #46 – Create Character in Dialogue (Cont.)

- Create a new FlowerDialogue script, write the following



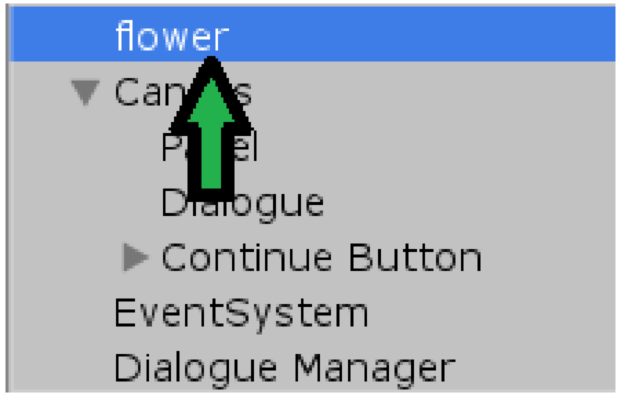
```
public class FlowerDialogue : MonoBehaviour {
    public Dialogue dialogueManager; //a variable that stores the Dialogue script that is attached to the Dialogue Manager gameobject

    // Use this for initialization
    void Start () {

    }

    // Update is called once per frame
    void Update () {

    }

    void OnTriggerEnter2D(Collider2D other)
    {
        if (other.tag == "Player") //if the player is the one that triggers the collider, then
        {
            string[] dialogue = { "Flower: Sonic, wait! Take a break and drink this before you proceed.",
                                  "Sonic: What's goin' on?",
                                  "Flower: Eggman and his new robot and thrashing the town!",
                                  "Flower: Knuckles and the others are tryna take 'im down.",
                                  "Flower: Now take this honey - it'll  restore your stamina!",
                                  "Sonic: Thanks! I'm on it!"}; //specify the dialogue between the player and the character(flower)
            dialogueManager.SetSentences(dialogue); //set the sentences array in the Dialogue script to above array
            dialogueManager.StartCoroutine(dialogueManager.TypeDialogue()); //start the coroutine of TypeDialogue(), which in turn starts the dia
        }
    }
}
```

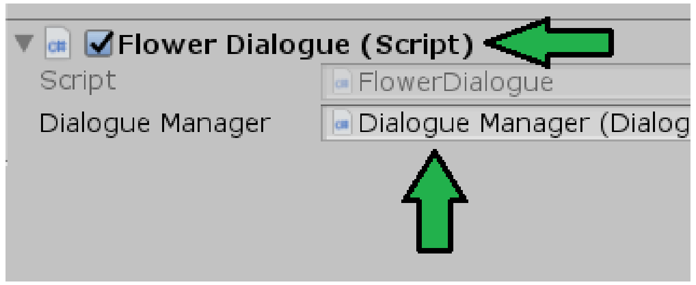
Exercise #46 – Create Character in Dialogue (Cont.)

- Select the flower

- Attach the FlowerDialogue script

- Drag and drop the Dialogue Manager gameobject onto the dialogue manager variable





Flower Dialogue (Script)

Script	FlowerDialogue
Dialogue Manager	Dialogue Manager (Dialog

Useful References:

- **Simple Dialogue System Tutorial. URL retrieved from:** https://www.youtube.com/watch?v=f-oSXg6_AMQ