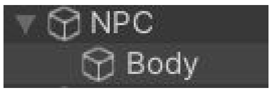
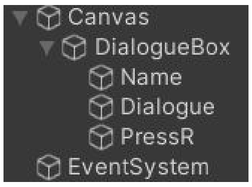
Tutorial 3: Dialogue System

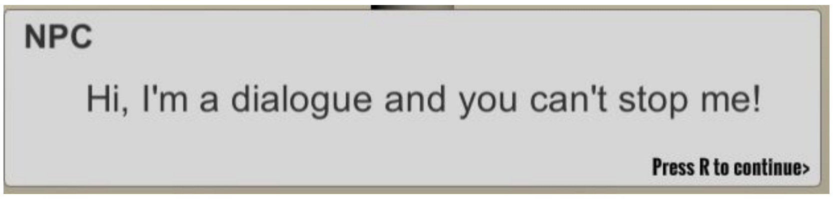
Preparation:

- You will need an NPC. In my case I used a cylinder to represent it.



- A canvas with a dialogue box, the name of the NPC, and the dialogue. I also added an indication on how to continue to the next sentence.





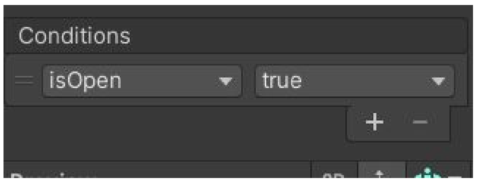
NPC

Hi, I'm a dialogue and you can't stop me!

Press R to continue>

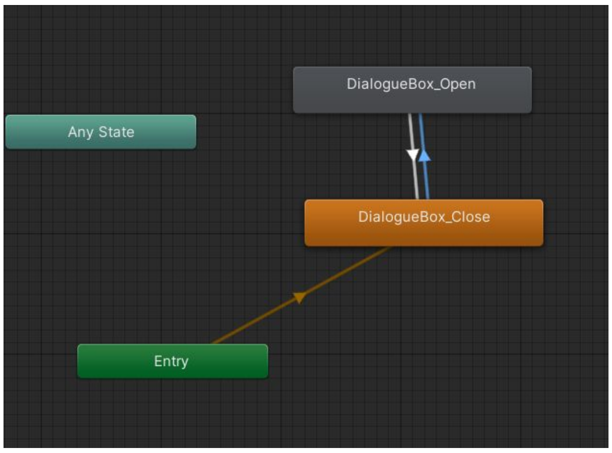
- An empty GameObject to later place a script in it. I've named it dialogue manager.

- Create two animations, one to open the dialog box and other to close it. Then, make transitions between them, make a new bool, "isOpen", and set the conditions for the transitions to happen. To close the box, the bool should be false and to open it, it should be true.



Conditions

isOpen true



- Create 3 scripts:

  DialogueManager: into the empty GameObject with the same name.
  Dialogue.
  DialogueTrigger: into the NPC.

Code:

Dialogue

In the line before the public class, write

[System.Serializable]

And, in the proper line of the public class, erase

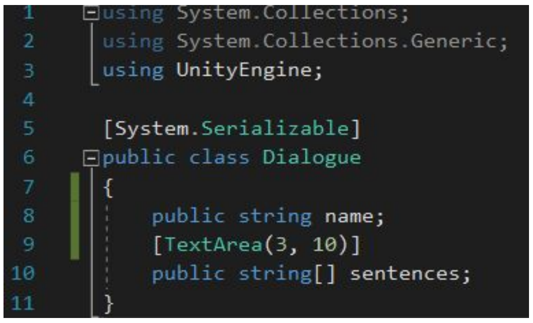
: MonoBehaviour

Both of these things are necessary because we are going to use this script as an object and we are going to pass it to the dialogue manager to pass the information.

It's not necessary the Start nor the Update functions for this script.
Create a variable for the name of the NPC and a string array where we are going to store the sentences. You can also increase the space given to write sentences with the second line.

```
public string name;
[TextArea(3, 10)]
public string[] sentences;
```

This is how it should look:



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

[System.Serializable]
public class Dialogue
{
    public string name;
    [TextArea(3, 10)]
    public string[] sentences;
}
```

DialogueManager

First of all, we need to set all the variables. The next variables are for the animation, for the text of the name of the NPC and of the dialogue, for the queue of sentences and for a bool that indicates that the dialogue has started.

```
public Animator animator;

public Text nameText;
public Text dialogueText;

public Queue<string> sentences;

private bool firstSentence;
```

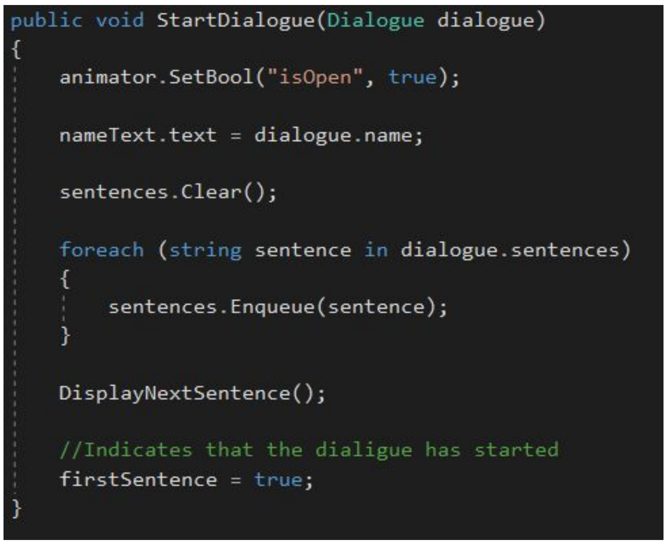
Inside the Start void first, set the last bool as false, and the write:

```
sentences = new Queue<string>();
```

Inside the Update void, set that a new function activates when you press R, or any letter that you want, and the dialogue has already started. In this case a function that makes the dialogue keep going.

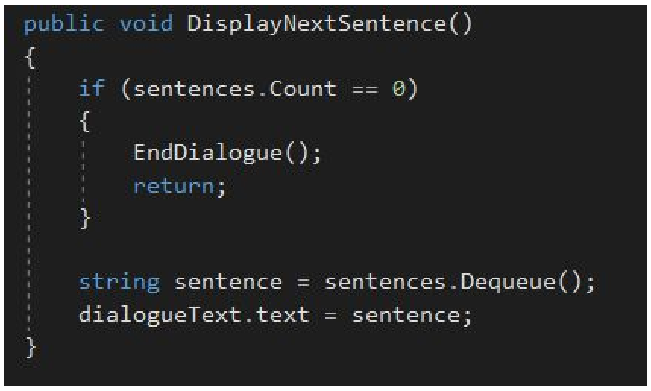
```
if (Input.GetKeyDown(KeyCode.R) && firstSentence)
{
  DisplayNextSentence();
}
```

Before creating the function DisplayNextSentence(), create one that starts the dialogue. Inside it set the bool for the animation and for the beging of the dialogue as true. The function should look like this:



```
public void StartDialogue(Dialogue dialogue)
{
    animator.SetBool("isOpen", true);

    nameText.text = dialogue.name;

    sentences.Clear();

    foreach (string sentence in dialogue.sentences)
    {
        sentences.Enqueue(sentence);
    }

    DisplayNextSentence();

    //Indicates that the dialigue has started
    firstSentence = true;
}
```

After that, create DisplayNextSentence(), in which you must set that if there are no more sentences the dialogue has come to an end and, if there is another sentence, that the text from the box must change.



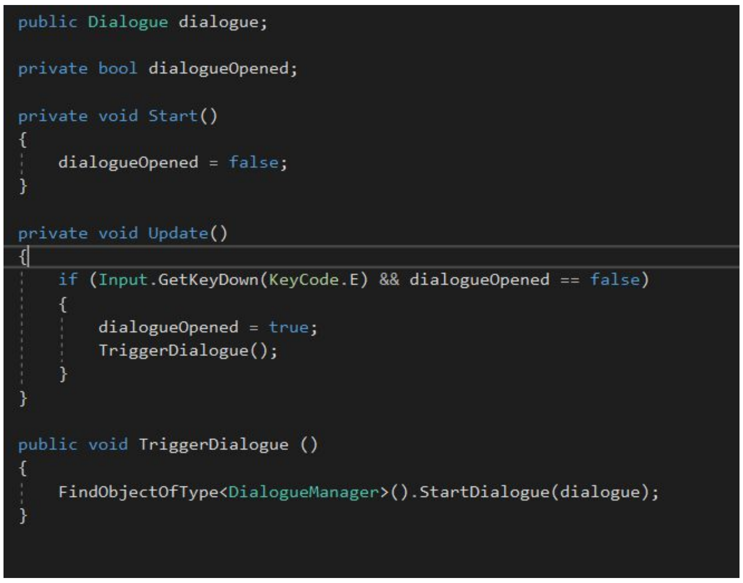
```
public void DisplayNextSentence()
{
    if (sentences.Count == 0)
    {
        EndDialogue();
        return;
    }

    string sentence = sentences.Dequeue();
    dialogueText.text = sentence;
}
```

As you see in the last screenshot, you need to create a function to end the dialogue. For that, just set the bool of the animation as false and the box of dialogue will go away.

```
void EndDialogue()
{
  animator.SetBool("isOpen", false);
}
```

DialogueTrigger

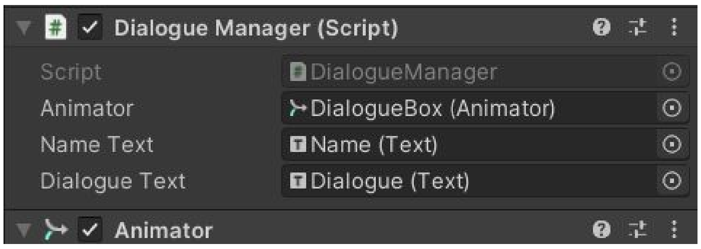
This last script is for activating the dialogue. Firstly, the variables should be for a dialogue ant the other for a bool that signals if the dialogue box has been opened or not. Then in the Start void, the bool should be set as false. In the Update void there should be a code to activate the dialogue when a key is pressed.



```
public Dialogue dialogue;

private bool dialogueOpened;

private void Start()
{
    dialogueOpened = false;
}

private void Update()
{
    if (Input.GetKeyDown(KeyCode.E) && dialogueOpened == false)
    {
        dialogueOpened = true;
        TriggerDialogue();
    }
}

public void TriggerDialogue ()
{
    FindObjectOfType<DialogueManager>().StartDialogue(dialogue);
}
```

If the dialogue is triggered, this code should be able to communicate order of activation with the other script.

Finally, there are some aspects of the Unity inspector that should be fixed.

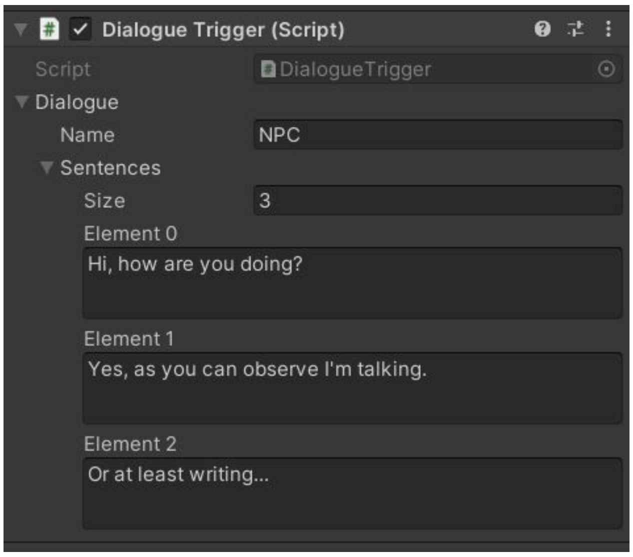


Dialogue Manager (Script)

Script	DialogueManager
Animator	DialogueBox (Animator)
Name Text	Name (Text)
Dialogue Text	Dialogue (Text)

Animator

You should fill any gap that might be empty in the inspector with the objects that you already should have available in your scene.



Dialogue Trigger (Script)

Script	DialogueTrigger
Dialogue	
Name	NPC
Sentences	
Size	3
Element 0	Hi, how are you doing?
Element 1	Yes, as you can observe I'm talking.
Element 2	Or at least writing...

This tutorial has been inspired by the next video:
https://www.youtube.com/watch?v=_nRzoTzeyxU&ab_channel=Brackeys