

Easy NPC documents

An example project is already set up in EasyNPCs/Scenes/ForStore

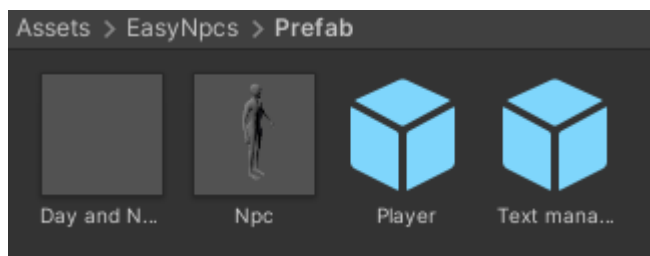
This document will provide you with information on using 'Easy Npcs' for your project. The main feature of the asset is the 'Npc' prefab stored inside EasyNpcs/Prefab/Npc

1. Setting up Easy npcs

1.1 Setting up

Before we get started we need a plane for the npcs to walk on and a navmesh configured on the plane so that the npcs can walk around.

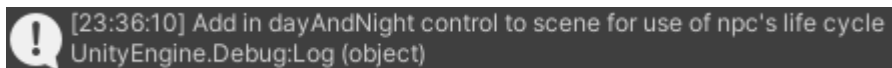
Afterward, go to EasyNpcs/Prefab and find the 'Npc' prefab.



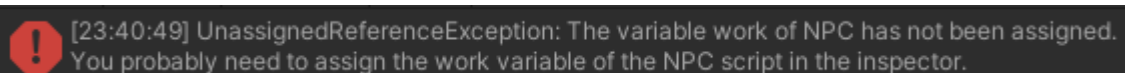
Click and drag the 'Npc' prefab into the scene.

1.2 Creating day and night cycle of npcs

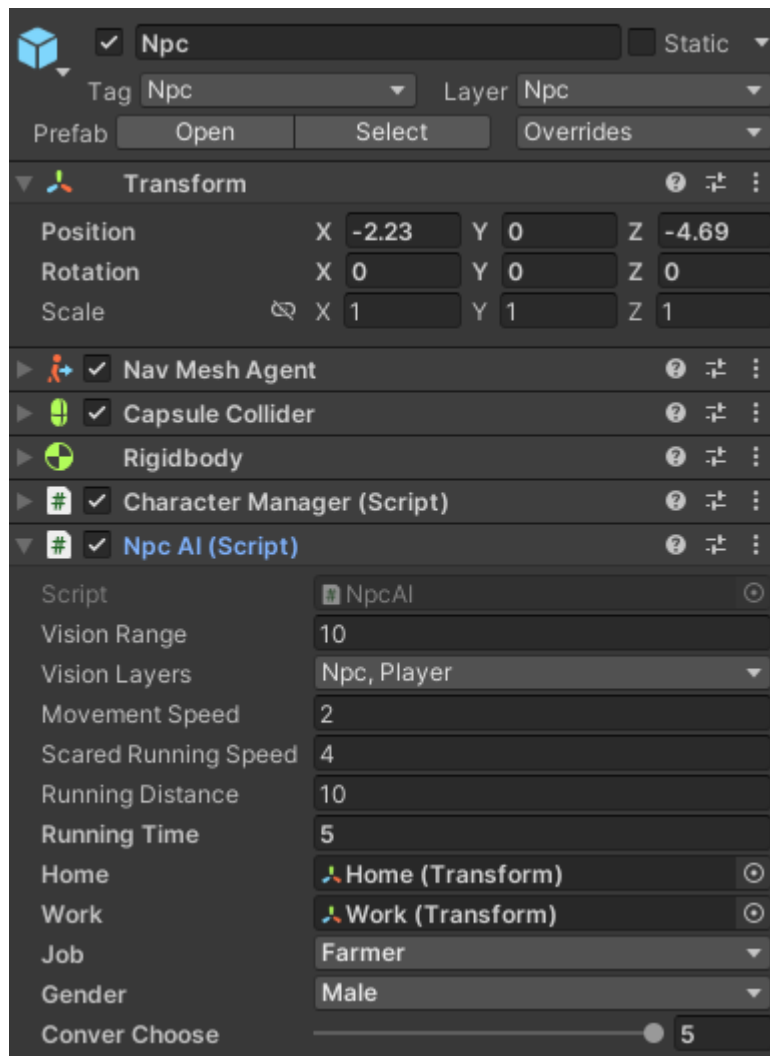
If you play the game you will get a debug.Log inside the console indicating that you need a day and night controller.



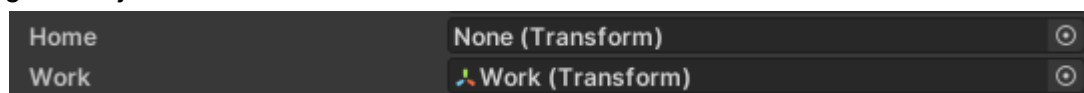
Inside EasyNpcs/Prefab and next to the 'Npc' prefab, you'll see the 'Day and Night Controller' prefab. Click and drag the prefab into the scene and play it again. This time you will get this error.



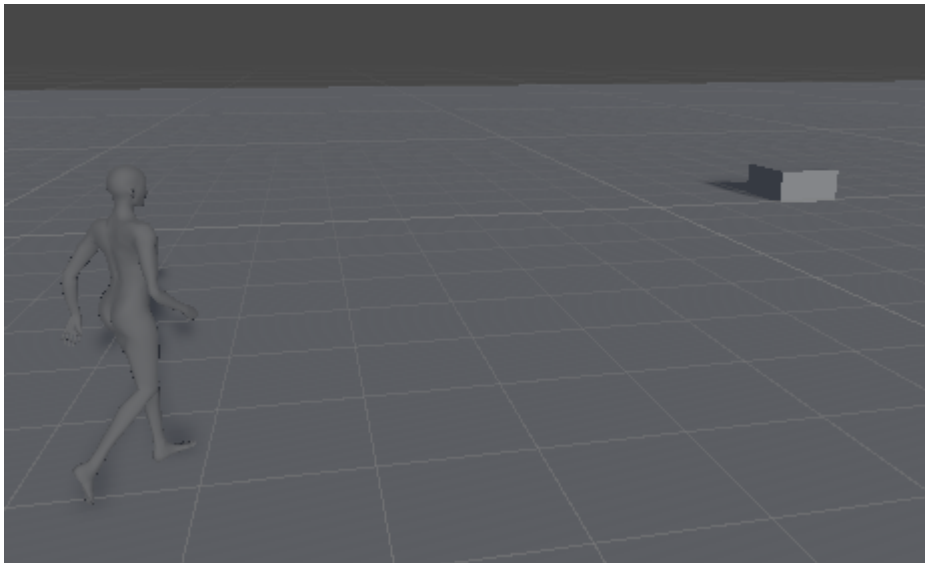
This means that the work position where the npc goes for the day is not assigned. Create a new game object named 'work' and place it on the plane. Click the npc we put in and go to the inspector.



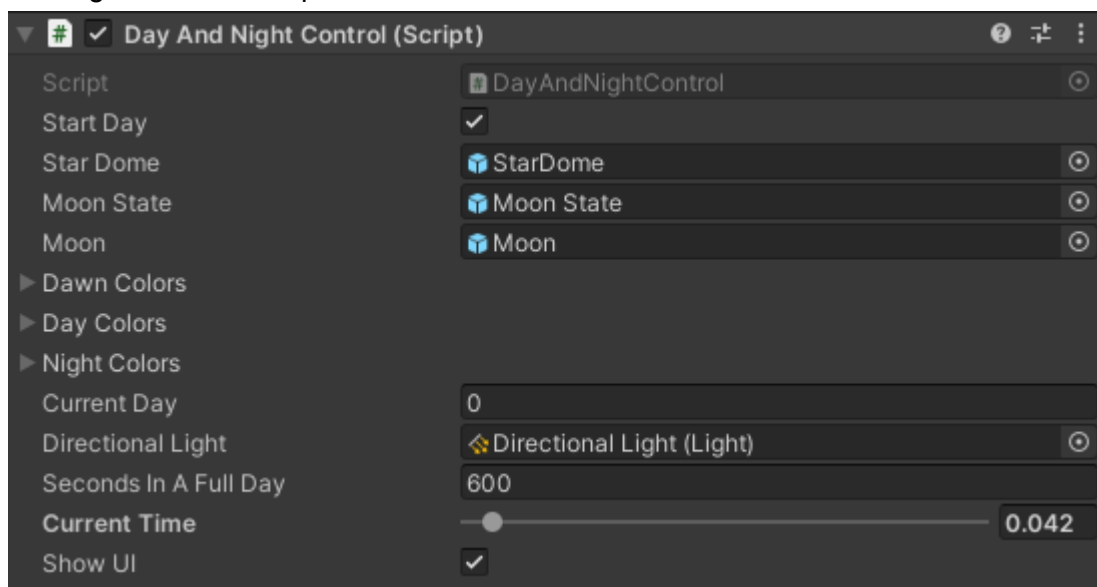
Find the 'Npc AI' component like in the picture. Find the variable 'Work' and assign the 'work' game object's transform to it.



Now play the game again and the npc will go to the designated spot.



You will see that the npc heads over to the position of the 'work' game object. Since the npc goes to work for the day, now we want it to go home for the night. Click on the day and night controller inside the scene and head to the inspector. You will find the 'Day and Night Control' component attached.

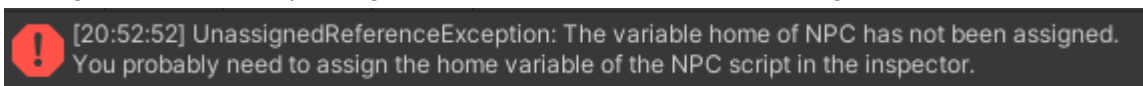


Notice the 'Current Time' variable.

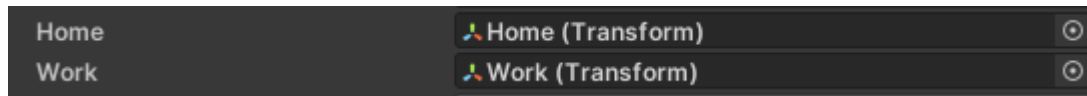


This variable indicates what time of the day it is. 0.3 ~ 0.8 is considered for morning and daytime. 0.8 ~ 1, 0 ~ 0.3 is considered as night time. When the variable reaches 1 it will go back to 0.

Change the time of day to night and check the console. You will get this error.



This time create a game object named 'home' and place it on the plane away from the 'work' game object. And assign the transform of the 'home' game object to the 'Home' variable inside the 'NPC' component.



Now once you hit play and set the time to night, the npc will move the location of the 'home' game object.

1.3 Assigning work

We can assign what kind of behaviors the 'NPC' will do when the npc reaches the 'work' position.

You can see an example in one of the scenes named 'AnimRigging'. If you go inside the scene you will see that the 'npc' has a component named 'shopkeeper' attached to it.

The 'shopkeeper' component drives from an interface named 'Iwork'.

```
Unity Script (1 asset reference) | 0 references
public class ShopKeeper : MonoBehaviour, IWork
{
    public RigBuilder rigBuilder;
    public TwoBoneIKConstraint right;
    public TwoBoneIKConstraint left;

    public GameObject rightPlacement;
    public GameObject leftPlacement;
    public GameObject shop;

    Rotate rotate;

    Unity Message | 0 references
    private void OnEnable()
    {
        rotate = gameObject.AddComponent<Rotate>();
        rotate.RotateTo(shop);

        right.data.target = rightPlacement.transform;
        left.data.target = leftPlacement.transform;
        rigBuilder.Build();
    }

    Unity Message | 0 references
    private void OnDisable()
    {
        right.data.target = null;
        left.data.target = null;
        rigBuilder.Build();

        Destroy(rotate);
    }

    2 references
    public Behaviour GetScript()
    {
        return this;
    }
}
```

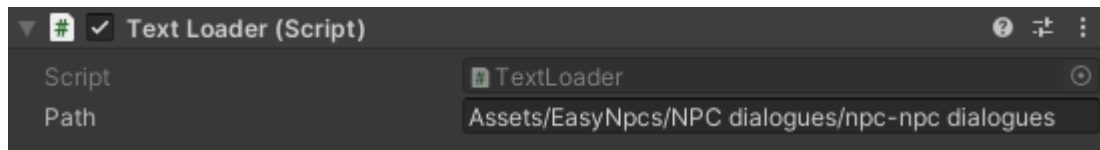
The 'npcAI' component will automatically find components derived from the 'Iwork' interface on start. Once found the script will be chosen as the work behavior script of the npc. And will be enabled when the 'npc' reaches its work position and will be disabled once it is not in the 'work' state.

2. Creating conversations

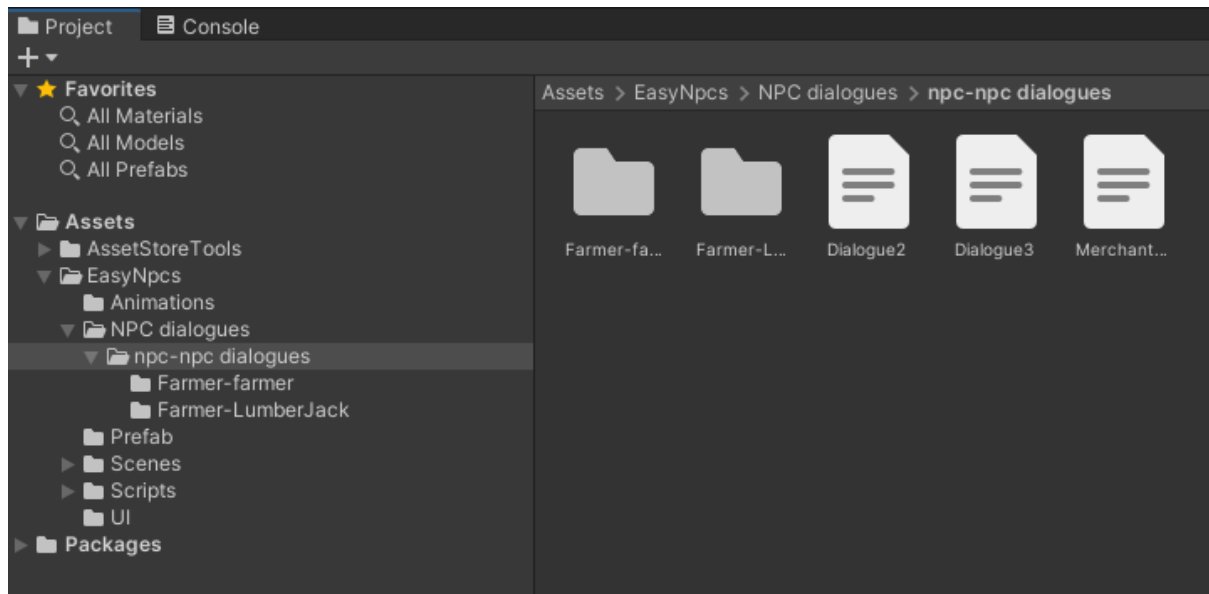
2.1 Adding Text manager

Before we can have the npc talk we need a text manager that loads the conversation files. You can find the 'text manager' prefab in EasyNpcs/Prefab. Click and drag the prefab into

the scene. If you see the text manager in the inspector you will see the 'text loader' component. With a string variable named path



Path is the path to where all the conversation files are stored. By default, it is EasyNpcs/NPC dialogues/npc-npc dialogues. If you head over to the files you'll see a list of conversations in 'txt' format.

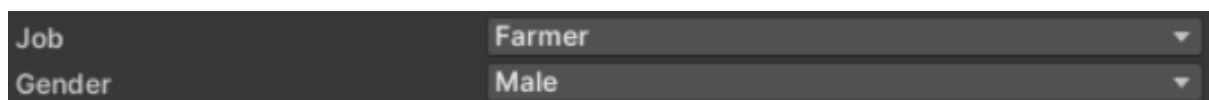


2.2 Trying out the conversations

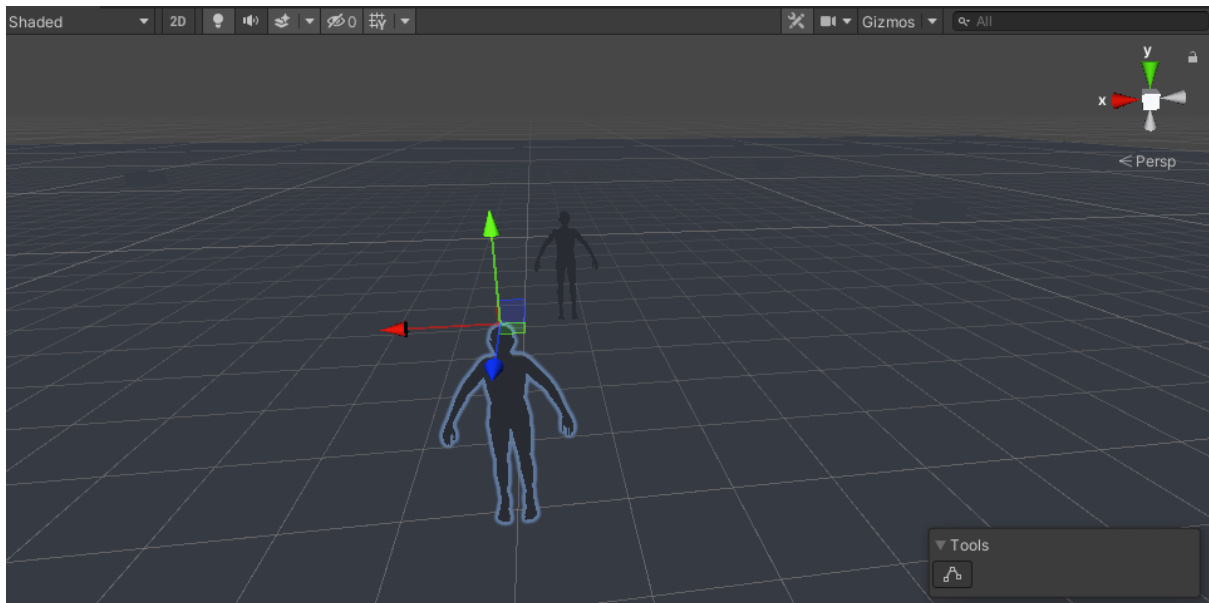
Let's try out the conversations. Go to the 'NPC' component of the npc we previously added in the scene to try out the day and night cycle. And from there find the 'Conver Choose' variable.



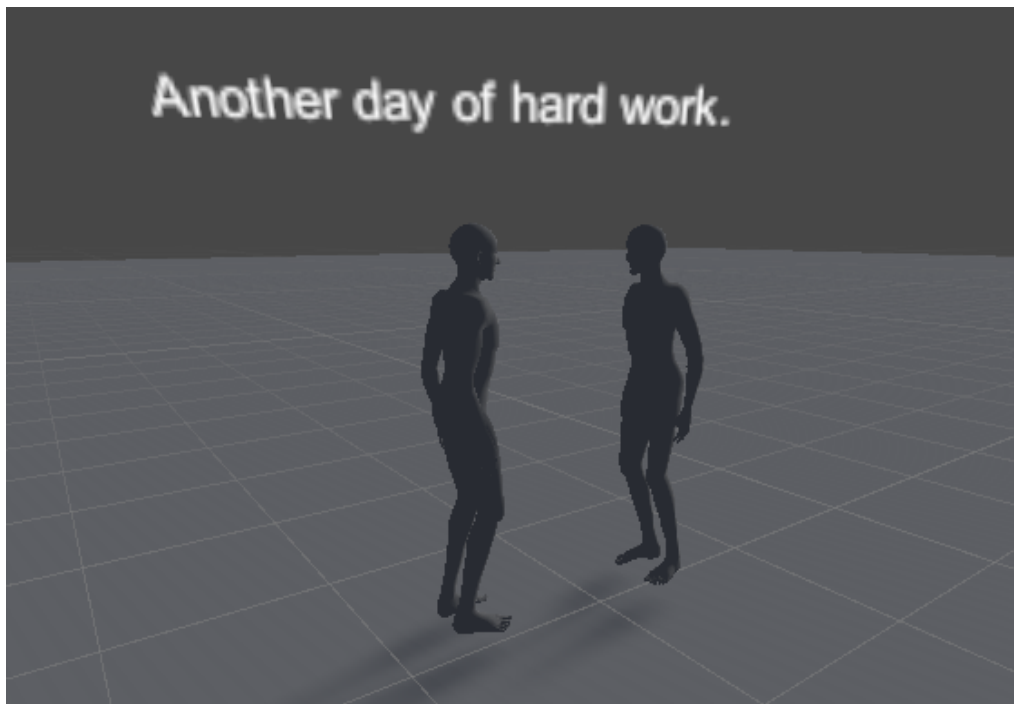
The 'Conver Choose' variable chooses how likely the npc will have an interaction with another npc near it. It is calculated per frame. Set the variable to a number greater than 0. Then find the 'Job' and 'Gender' variables inside the 'NPC' component. Set each of the variables to 'Farmer' and 'Male'.



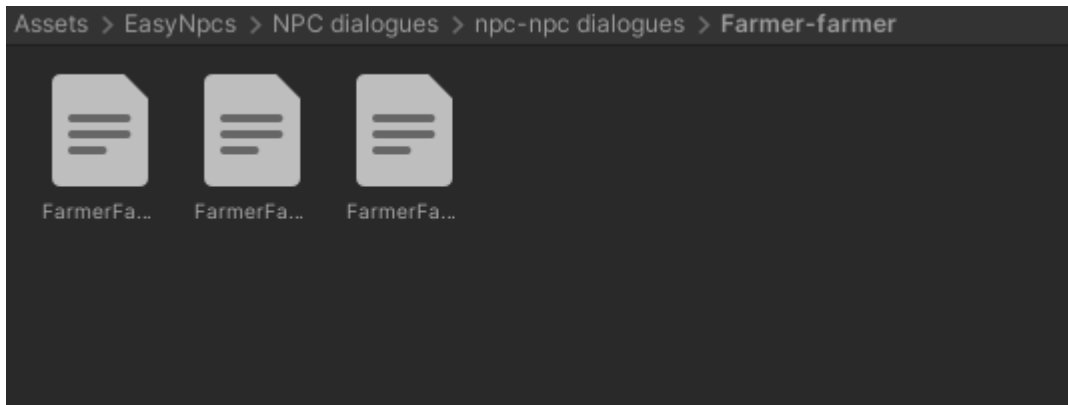
Add another npc to the scene. And just as before, set the 'Conver Choose' variable to a number bigger than 0. And set the 'Job' and 'Gender' variables to 'Farmer' and 'Male'.



Now run the game and the npcs will have conversations.



Go over to EasyNpcs/NPC dialogues/npc-npc dialogues/Farmer-farmer. The 'txt' files inside here are only for conversations between 2 male farmers. If you keep on running the conversations between the 2 npcs inside the scene you will notice that only the files inside 'Farmer-farmer' are chosen and executed.



Open the first text file from the left.

```
!Male
!Farmer
Hello
How are you doing?
Bye
!Male
!Farmer
Hi
I am fine
Goodbye
```

'!Male' and '!Farmer' are indicators that this section is to be said by an npc whose gender is male and whose job is being a farmer. The section ends before the indicator for the 2nd npc is written. So in this case, the section ends with the sentence 'Bye'. The second indicator and section are for the 2nd npc. So when this conversation is executed it will go by this:

Farmer1: Hello

Farmer2: Hi

Farmer1: How are you doing?

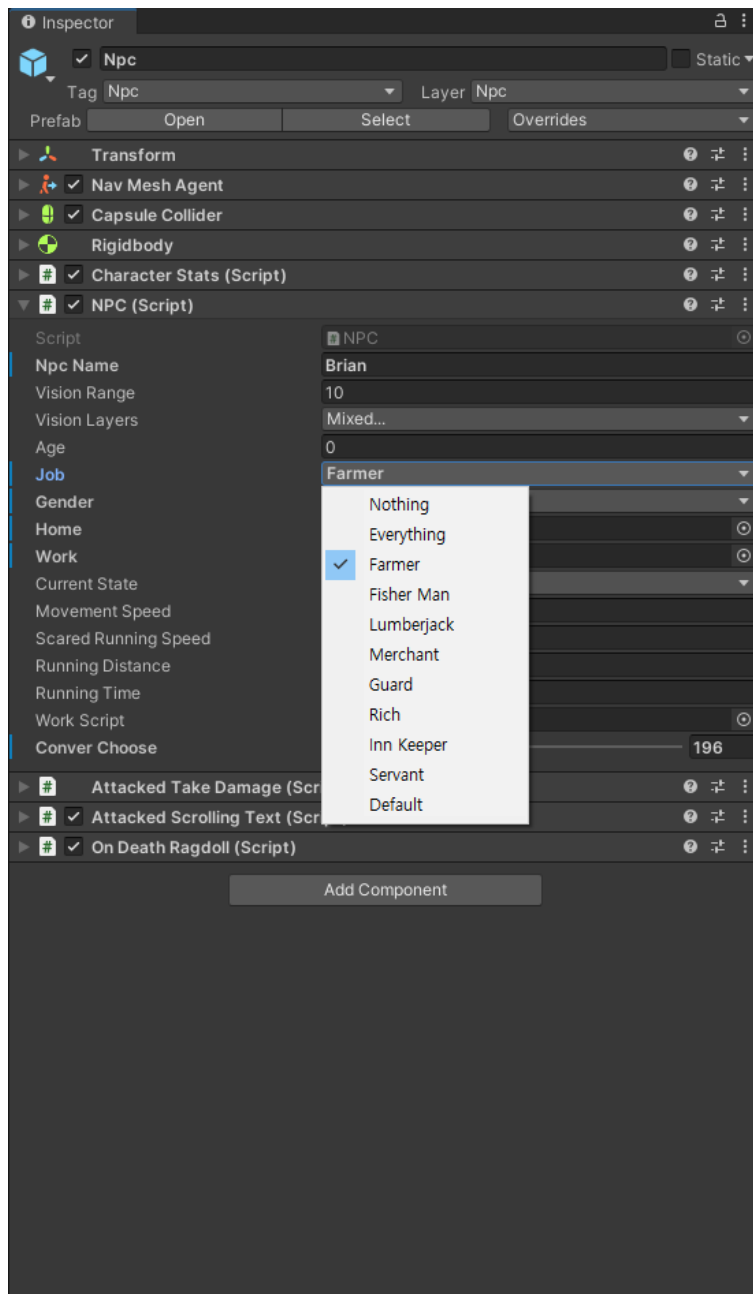
Farmer2: I am fine

Farmer1: Bye

Farmer2: Goodbye

2.3 Creating conversations

Let's write our own conversations. First, we need to see what types of jobs and genders are there. Go to the npc component by the inspector and click on the Job and Gender variables.



You'll be able to see all the available jobs and genders.

Let's write a conversation between a rich female npc and a male farmer npc. The conversation will go as the following

RichFemale: Do not come near me dirty pheasant.

MaleFarmer: What did you just say to me woman?

RichFemale: Ugh, nevermind you, dog.

MaleFarmer: Get out of here before I give you what's right for you.

Make a 'txt' file and write like the following

!Female

!Rich

Do not come near me dirty pheasant

Ugh, never mind you, dog.

!Male
!Farmer
What did you just say to me woman?
Get Out of here before I give you what's right for you.


Save it and put it inside EasyNpcs>NPC dialogues>npc-npc dialogues.
Warning! Do not leave an extra space of a line at the end. This is an example of a wrong 'txt' file.

```
1  !Female
2  !Rich
3  Do not come near me dirty pheasant
4  Ugh, never mind you, dog.
5  !Male
6  !Farmer
7  What did you just say to me woman?
8  Get Out of here before I give you what's right for you.
9
```

The asset will also take the blank space as a string which will result in an error. This is the right way to write the 'txt' file.

```
1  !Female
2  !Rich
3  Do not come near me dirty pheasant
4  Ugh, never mind you, dog.
5  !Male
6  !Farmer
7  What did you just say to me woman?
8  Get Out of here before I give you what's right for you.
```

Change one of the npc's jobs and gender to rich and female. And you will be able to see the npcs speaking the conversation we've written above.
But instead of having a successful conversation, the npcs might not have one and you might get this error.

 [17:03:40] ArgumentOutOfRangeException: Index was out of range. Must be non-negative and less than the size of the collection.
Parameter name: Index

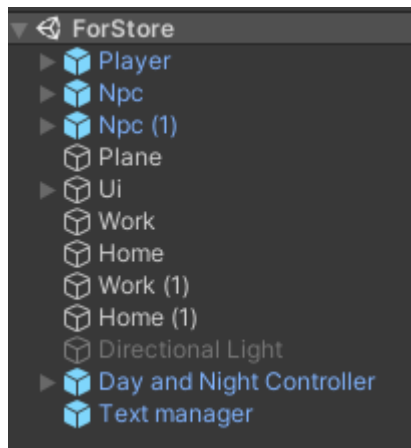
This is because each npc has its own specific Id. Depending on the Id, npcs can start a conversation with another npc, letting it start the conversation first. Sometimes because of the Id, an npc is not allowed to start a conversation with a specific npc. And is only allowed to receive a conversation. If you change the jobs and genders of the 2 npc to the opposite the npcs will be able to have a conversation.

3. Damaging npcs

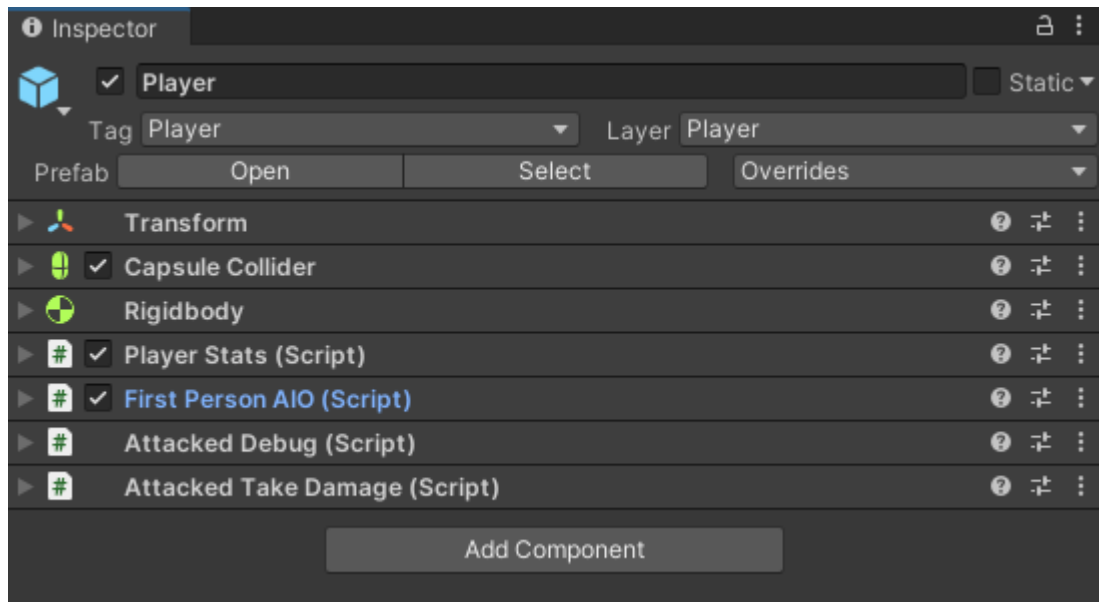
3.1 How to give damage to npcs

We will go through how the attacks are executed in Easy NPCs and how you can add them to your character.

If you head over to the sample scene(named 'For Store') we've already made you will be able to find the 'Player' game object.



Click the 'Player' object and check it inside the inspector. Find the 'First Person AIO' component.



Open the 'First Person AIO' component in a scripting API. Find the method `AttackTarget(GameObject target, bool bashAttack = false)`

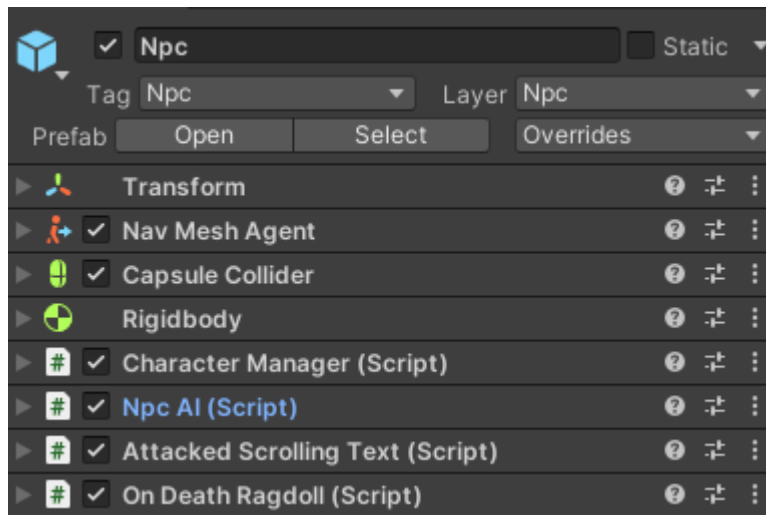
```
1 reference
void AttackTarget(GameObject target) //decides and creates attack on target
{
    Attack attack = new Attack(10);

    var attackables = target.GetComponentsInChildren(typeof(IAttackable)); //IAttackable has OnAttack() when executed player's attack
    foreach (IAttackable attackable in attackables)
    {
        attackable.OnAttack(gameObject, attack);
    }
}
```

This method is executed when the player presses mouse1 to attack.
'GameObject target' is the npc that the player is attacking.

Then we create an 'attack'. 'Attack' is a class that can be returned as an int when dealing damage. Right now it's written 'Attack attack = new Attack(10)'. This means we've created an 'attack' that will give 10 damage to the npc when it is used.

The next line is written 'var attackables = target.GetComponentInChildren(typeof(IAttackable)); Which means get all the components inside the target npc that is inherited from interface 'IAttackable'. The npc has 2 components that inherit from 'IAttackable'. Those are the 'NPC AI' and 'Attacked Scrolling Text' components. 'NPC AI' handles the damage while 'Attacked Scrolling Text' pops up a text that the npc has been attacked.



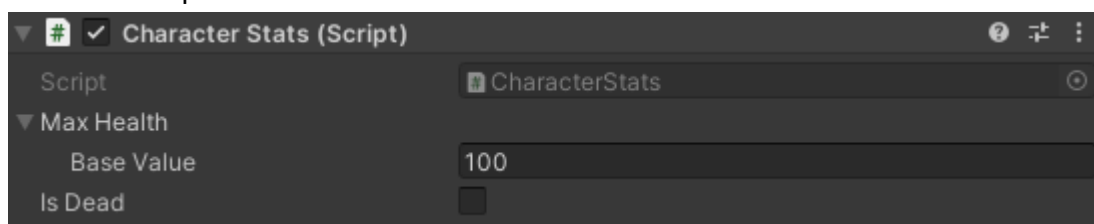
The interface 'IDestructable' has a method named OnAttack().

```
using UnityEngine;

10 references
public interface IDestructable
{
    6 references
    void OnAttack(GameObject attacker, Attack attack);

    6 references
    void OnDestruction(GameObject destroyer);
}
```

'CharacterStats' is the component that stores the current health rate of the 'NPC'. The 'NPC AI' changes the health rate when 'On Attack' is triggered. It is also responsible to show whether the npc is alive or dead.



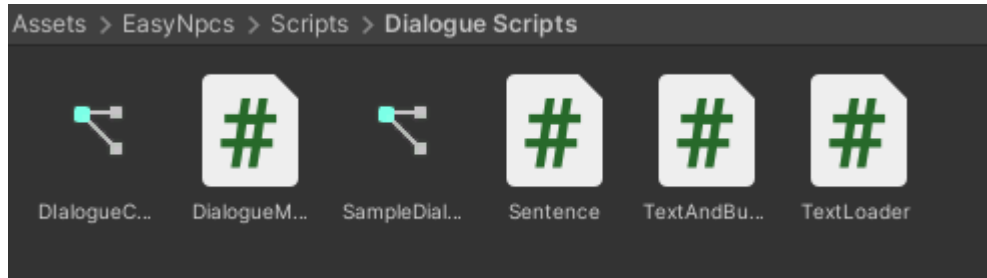
Max Health>Base value: Health of the npc on start.

Is Dead: Changes to true if npc's health has reached below 0. Otherwise false.

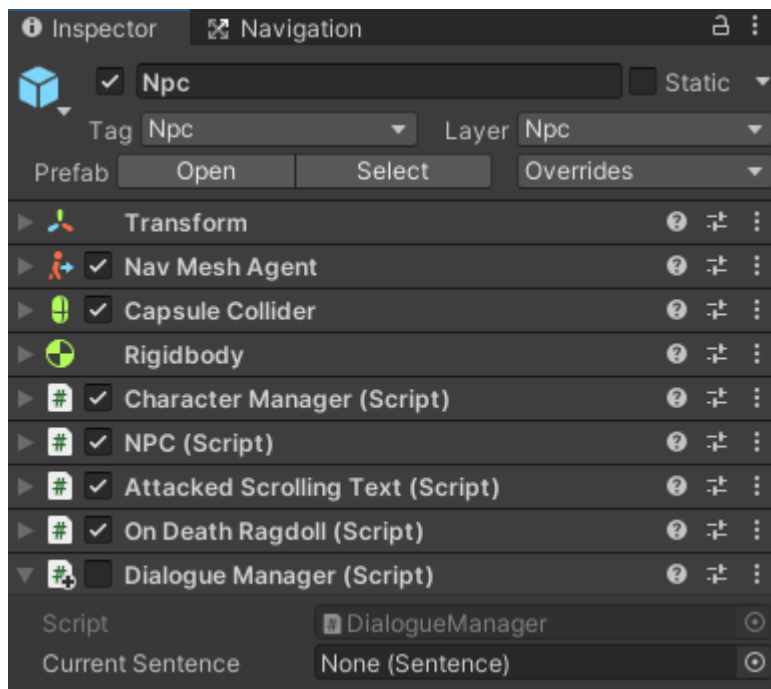
4. Dialogues

4.1 Adding the Dialogue Manager

To have the player talk with an npc, first, we need to add a 'Dialogue Manager' component to the npc. The 'Dialogue Manager' is located at EasyNpcs/Scenes/Scripts/Dialogue Scripts.

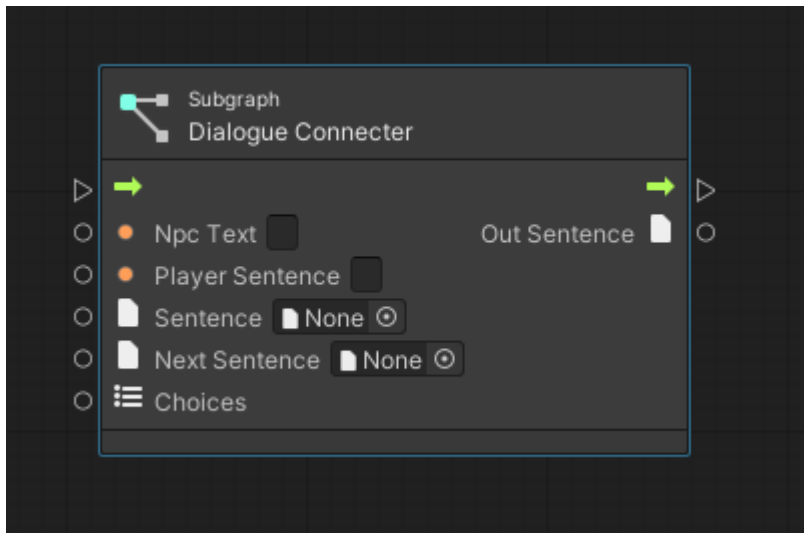


Add the component to an npc.



Make sure the 'Dialogue Manager' is disabled as default.

We now need to assign a dialogue to the 'Dialogue Manager'. We will use visual scripting to handle the branches of the dialogue. Create a new visual script and add the subgraph 'Dialogue Connector'. The 'Dialogue Connector' is placed next to the 'Dialogue Manager' inside the files.



Each of the variables means the following.

Npc Text: Text the npc will say.

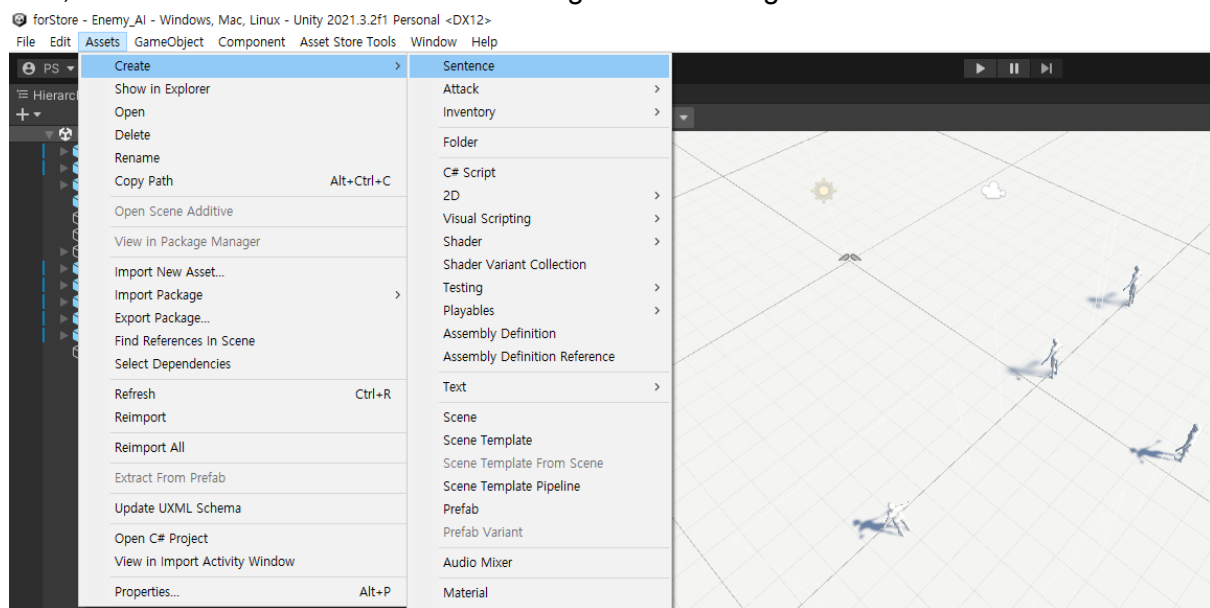
Player Text: Text the player will say.

Sentence: A data structure that contains the 'Npc Text' and 'Player Text'.

Next Sentence: Next sentence of the dialogue with no choices.

Choices: List of sentences that can be chosen as the next sentence.

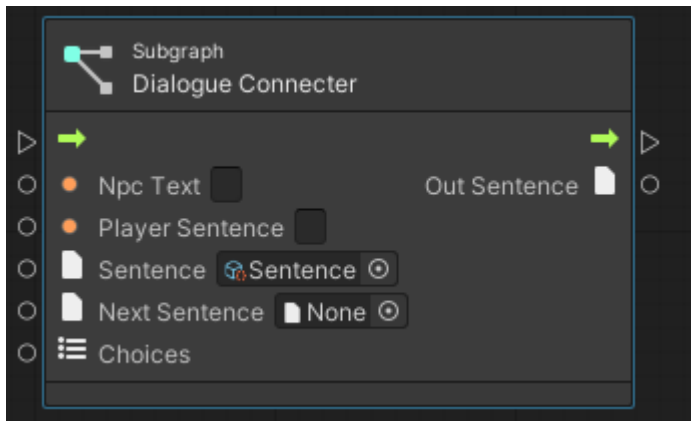
First, we need to create a 'sentence' to assign to the 'Dialogue Connector'.



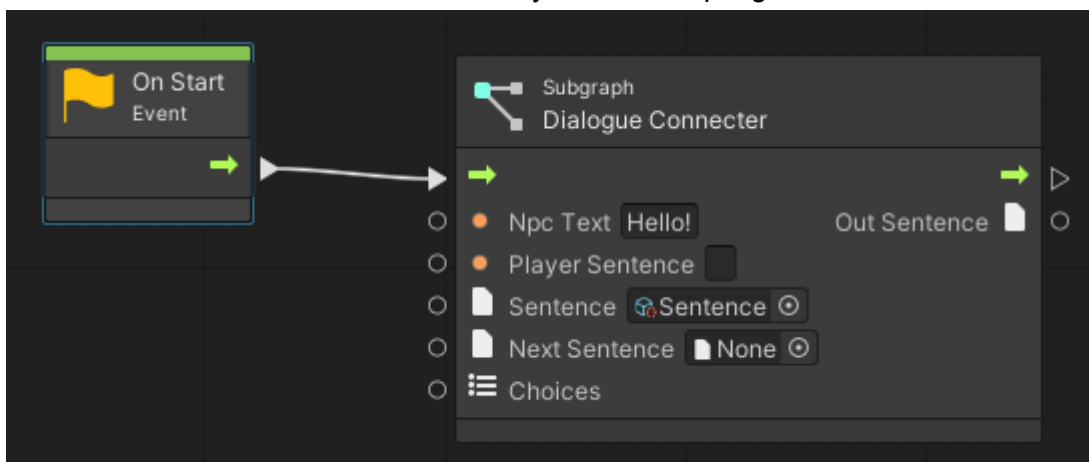
Go to Assets/Create and choose the 'Sentence' box and create a 'sentence'.



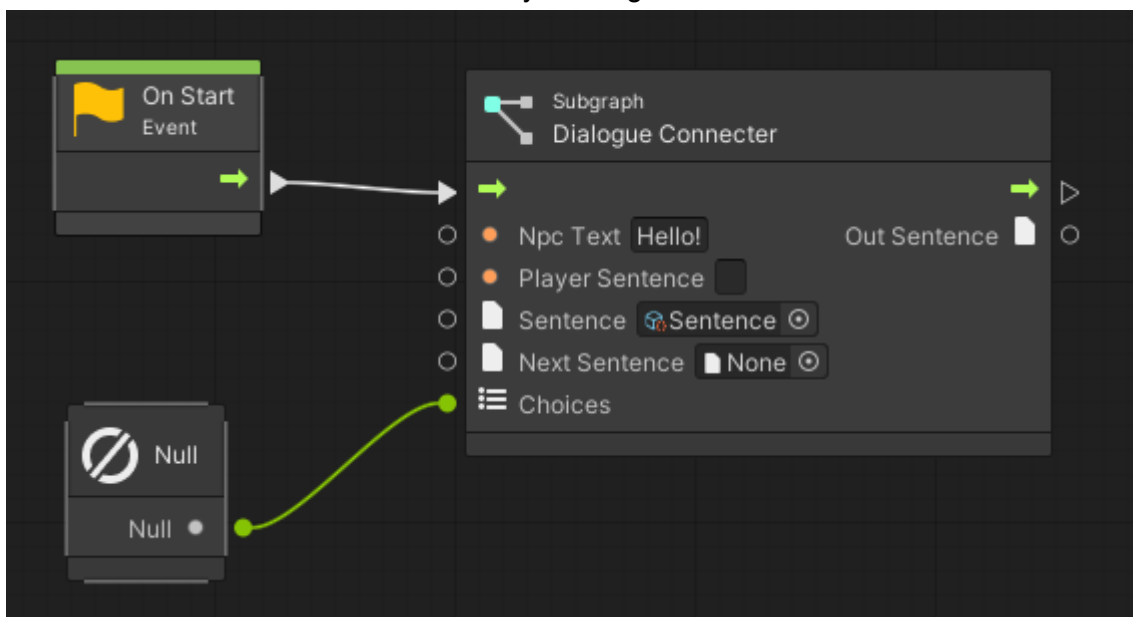
Assign the 'sentence' to the 'Dialogue Connector'.



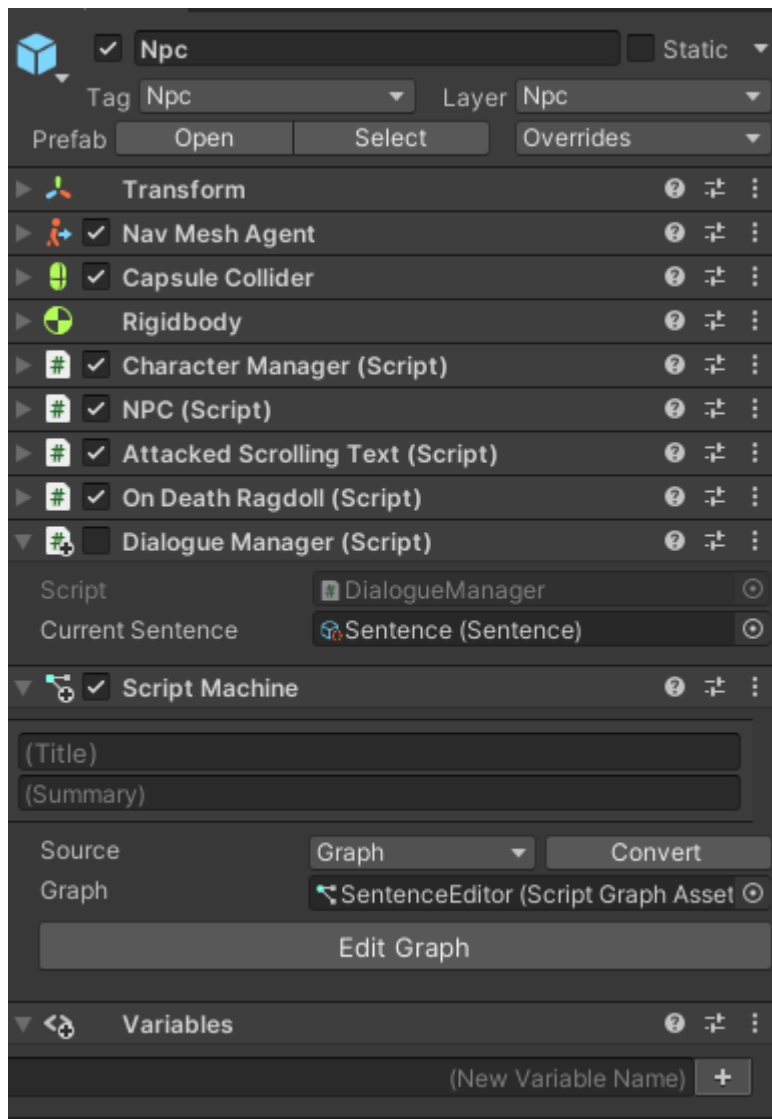
Then we add the sentence for what we want the npc to say. We'll go for "Hello!". Finally, we connect it with the start statement of Unity's visual scripting.



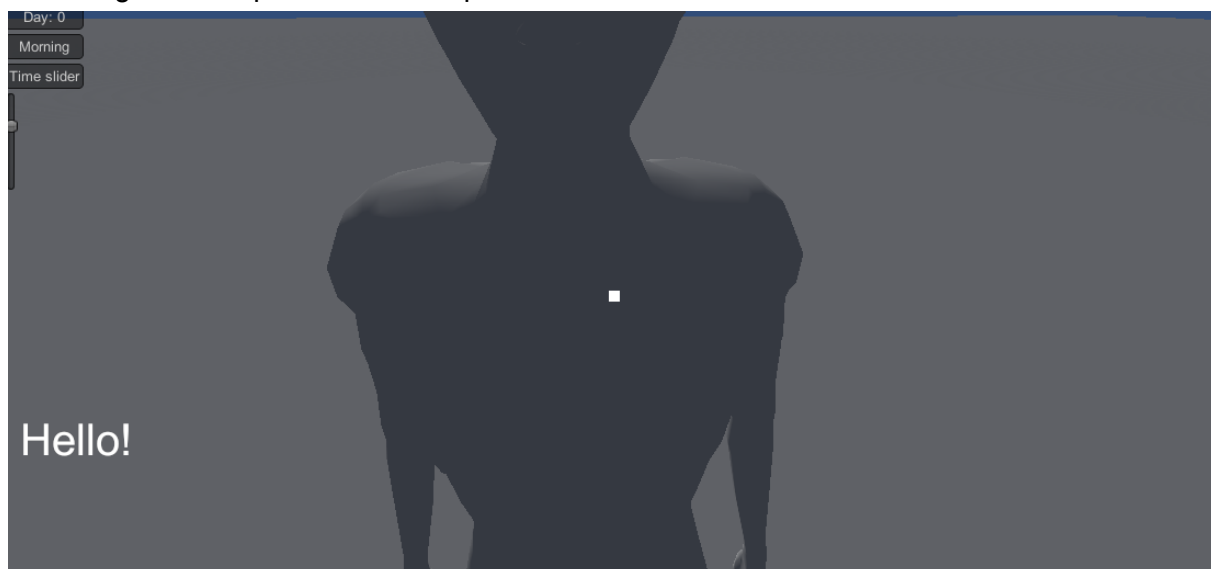
Make sure to set the 'Choices' to null or you will get errors.



Attach the visual script we made to the npc that already has a 'Dialogue Manager' attached.

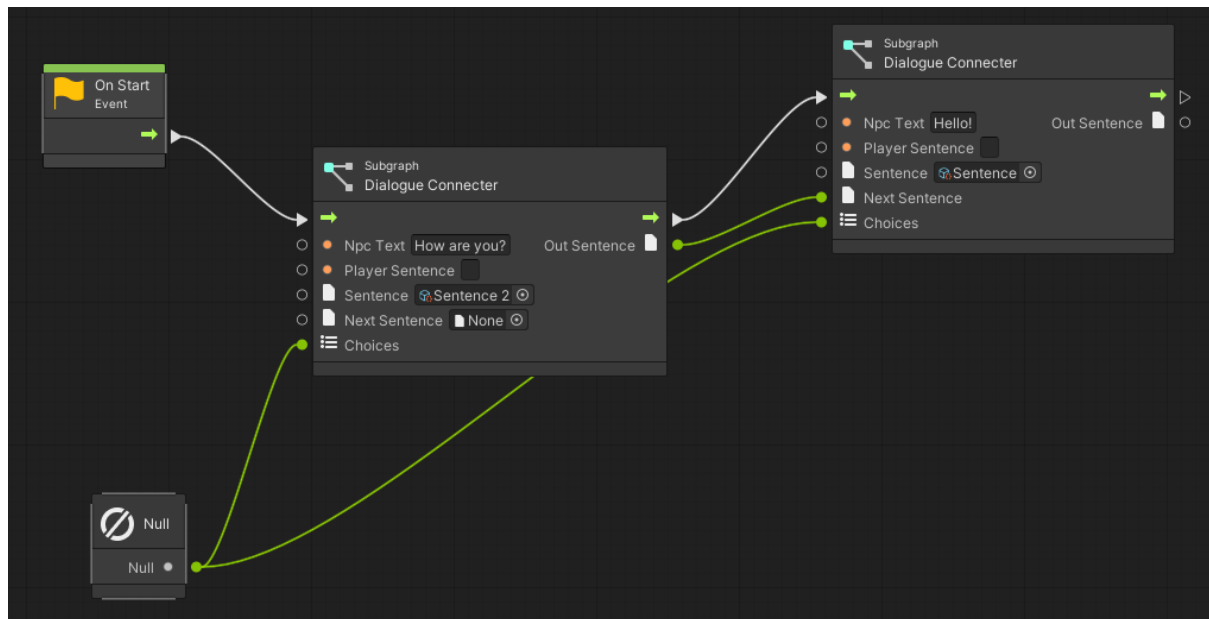


Run the game and press e on the npc we modified.



The npc will rotate towards the player and the UI will print out “Hello”. If we click on mouse button 1 the dialogue will be finished.

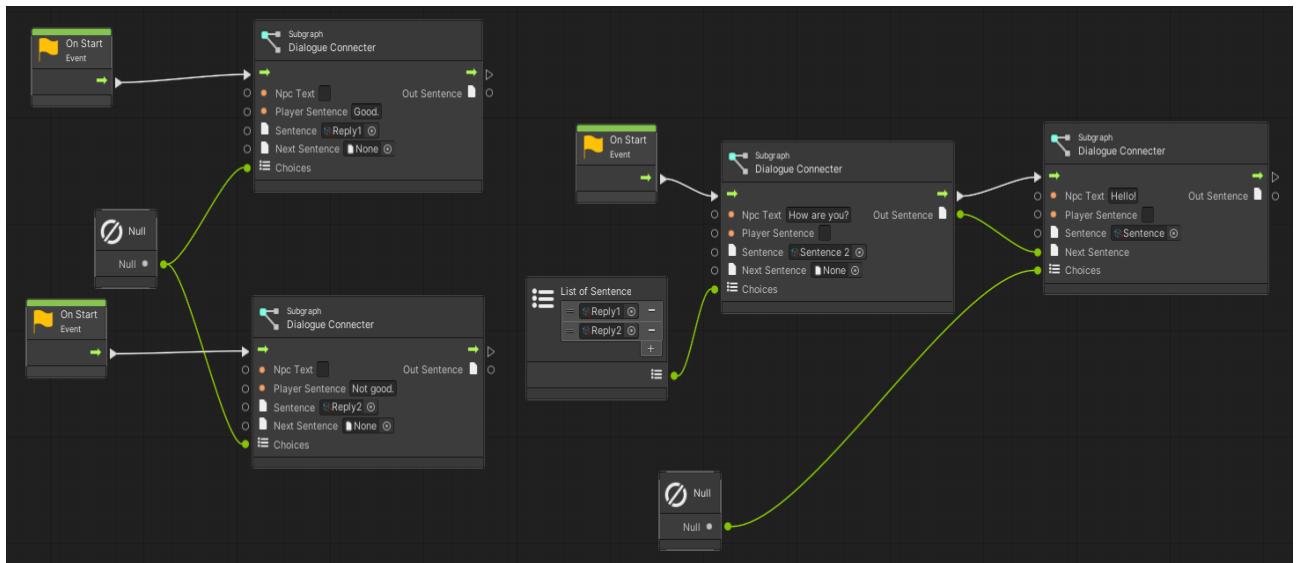
Now create a new sentence and place it inside a ‘Dialogue Connector’. Then assign it to the ‘Next Sentence’ variable on the previous dialogue we created.



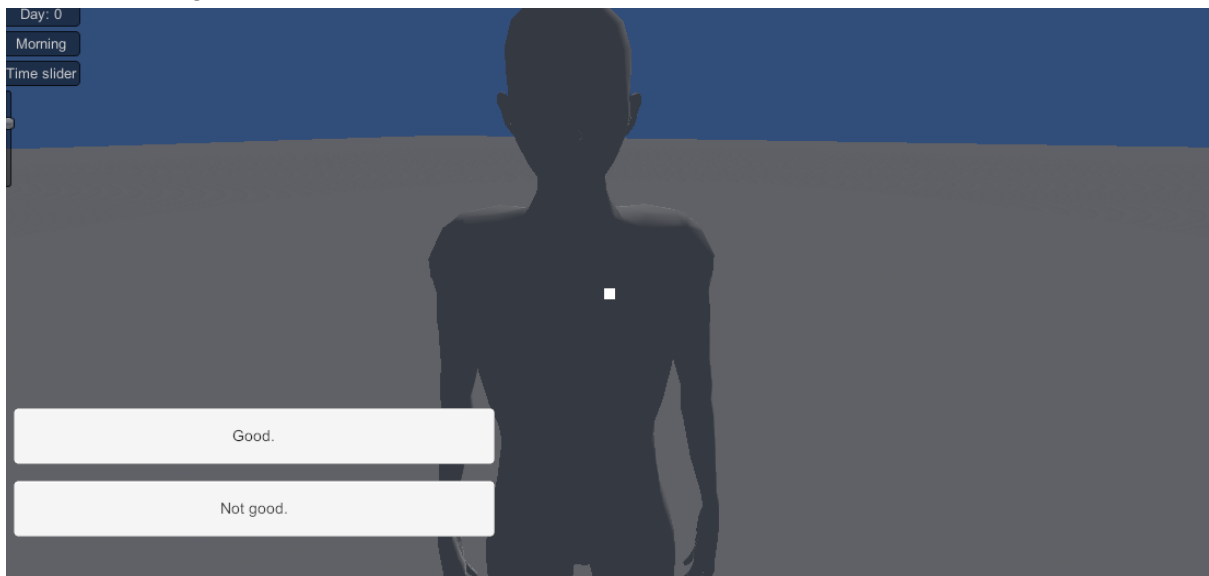
Run the game and talk to the npc again. This time once you press the mouse button1 the UI will print out “How are you?”.



You can exit the sequence again by pressing mouseButton1 after “How are you?” is printed. Now we’re going to give choices for the player to choose and branch out. First, create 2 more sentences and assign them to each of their own ‘Dialogue Connector’. Fill in the texts ‘Player sentence’ for the 2 choices the player can make. Then create a list of sentences and assign the 2 sentences to it. Connect the list to the ‘Choices’ variable of the ‘Dialogue Connector’ that contains “How are you?”.



Now after the UI prints out “How are you doing?” the player is able to make 2 choices that finish the dialogue.



You can connect sentences to the choice sentences if you want the npc to reply.

If you wish to intergrade Easy Npcs with your game to the full extent you will need to go through the source code.

If you wish to contact us directly: 1stmanleader@gmail.com

Also, Easy Npcs will be constantly updated. For we believe long-time support is as important as creating something new.