Introduction

Hello and thank you for purchasing EasyDialogue! I know you spent some hard earned money on this, so hopefully I can make the startup process easy!

What this package is aiming to do is the following:

- Give you an easy and intuitive graph-based system to create dialogue.
- An interface to ask for the current dialogue

That’s it! I had many earlier versions of this asset that aimed to provide more functionality, but limited the amount the user could choose where/how to display the text, and I ended up remaking this three times to get down to the simple system that I wanted. So that is important to know, currently in EasyDialogue, it does not do anything more than facilitate you creating and getting text for your game, it does **not** display text.

First things first, you need to know that the graph editor was built on top of another package called **XNode**. This is a free asset hosted on github and I recommend checking it out, however, if you are encountering issues with the graph editing, try and determine if the issue lies in the interaction with the EasyDialogue nodes, or if it has to do with the **XNode** editing system.

Let's move onto the terminology.

Terminology

EasyDialogue Graph

A dialogue graph is what we use to represent a conversation, you can string together sentences, as well as provide branching dialogue options that lead to various outcomes. You can also jump into dialogue from other graphs. Dialogue graphs support a "LocalGraphContext" in which you can provide your own script, with any variables, and any replacement functionality that you choose.

EasyDialogue Node

An EasyDialogue Node is the primary building block of all conversation in EasyDialogue, EasyDialogue Graphs are made up of these. They house Character information, text, and any player responses. As well as the nodes that come before and after it.

Jump Node

A jump node provides a way to make EasyDialogue nodes reusable, as you can jump back to them, even if they are from another graph. As a node that you are jumping to may have a different character set to it, the jump node also provides an override character setting, where you can force the character while jumping.

Character

EasyDialogue provides a class called "Character" that is referenced inside of the EasyDialogue Nodes, they provide an easy way to associate text to a character in-game, you can add on to the information in the Character class, however, by default it has an id, a name, and a sprite.

EasyDialogue Manager

This is the interface between the user and the EasyDialogue system. You will do all or almost all of your interactions with EasyDialogue through this manager.

Events

There are C# events that you can read about **here**.

The Basics of C# Events

Basically this is what they mean. I can, in code, “Invoke” a function similar to this, “OnDialogueStarted?.Invoke(_dialogueGraph, _dialogueLine);” All this invocation does is call many functions, it calls the functions of whoever wants to be called, and matches the parameters. So functions “Subscribe” or “Listen” to this event, and then when it is invoked, those functions are called.

For example. I have a UIManager, I want the dialogue box to appear when the dialogue starts. So, I can “Subscribe” a function that toggles the UI to the event that is invoked when dialogue starts.

Easy Dialogue Events

EasyDialogueManager.OnDialogueStarted

Parameters:
EasyDialogueGraph, dialogue_line

Description:
This event is invoked when the dialogue starts.

EasyDialogueManager.OnDialogueProgressed

Parameters:
EasyDialogueGraph, dialogue_line

Description:
This event is invoked when the dialogue progresses to the next node.

EasyDialogueManager.OnDialogueEnded

Parameters:
EasyDialogueGraph

Description:
This event is invoked when the dialogue ends.

Getting Started

Sample Scene (Minimal_Implementation):

To get started with EasyDialogue, it would be best to check out the sample scene, you will see that there are many characters and EasyDialogue graphs created already. If you hit Play, you can see the system working by clicking ‘N’ to progress the dialogue forwards. If you click into the Dialogue Graphs in the Project view, and double click on the “Intro” graph, it will open in the XNode window for you to see how everything is set up.

You can click-and-hold the middle mouse button to drag around the scene, scroll to zoom, left click to select, and right click on a node to bring up node options, or right click in empy space to place a new node. You can interact with the nodes as you would the unity inspector. You can click and drag from the circles at the end to move around the node connections.

To get familiar with the interfacing with EasyDialogueManager please click on the DialogueManager in the scripts folder. Review this code and any comments to get a feel for the EasyDialogue manager. Feel free to also open up the EasyDialogueManager to see what is going on in the code if that helps you at all!

New Scene:

When creating a new scene and you want to get started with EasyDialogue, there are three things you have to do.

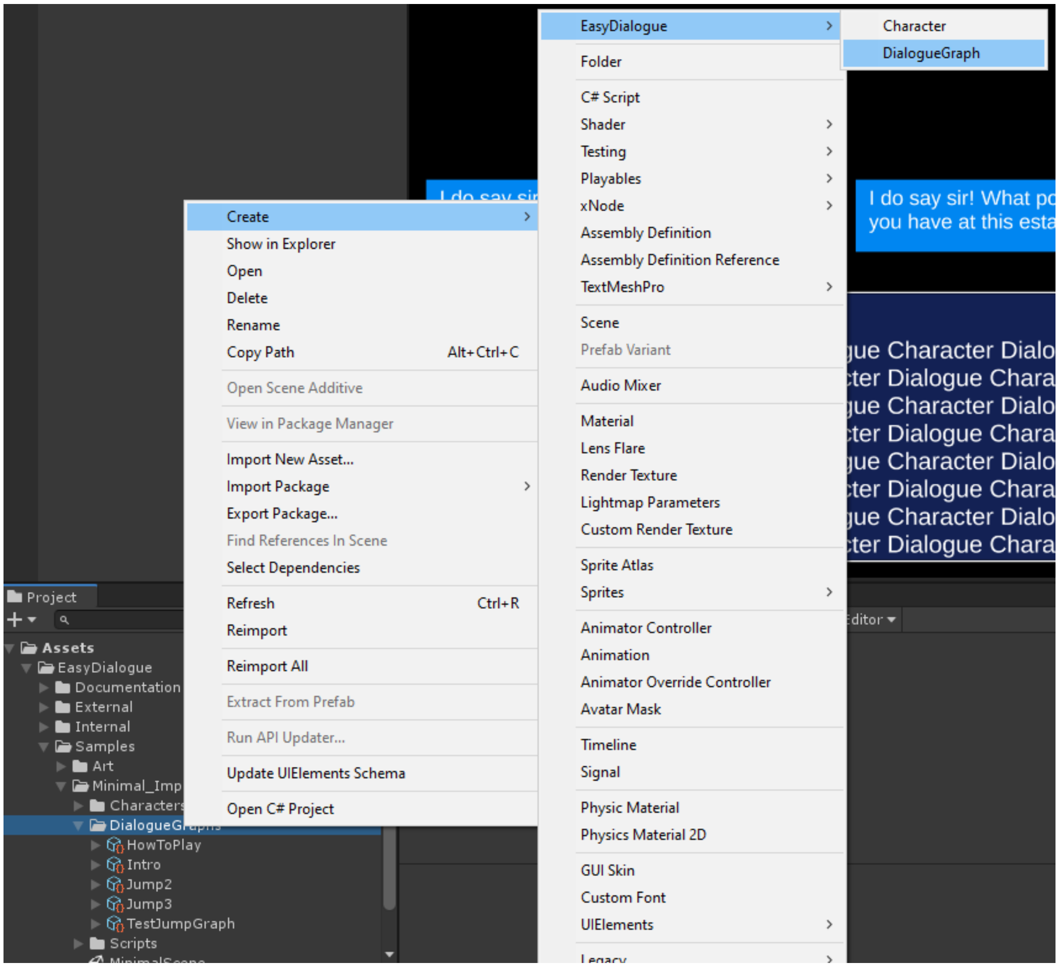
1. Add the EasyDialogueManager component onto a gameobject.
2. Create an EasyDialogue graph to display dialogue
3. Create a script to interface with the EasyDialogueManager.

1. Adding EasyDialogueManager Component:

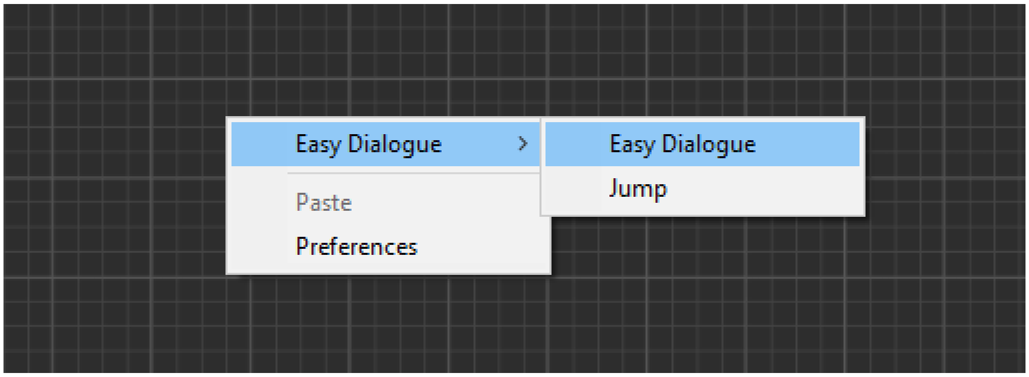
To add this component, simply click on a gameobject, and click Add Component, and type in EasyDialogueManager.

2. Creating an EasyDialogue Graph:

Go to your Project view, and find where you would like to save the graph. Right click and navigate to Create->EasyDialogue->DialogueGraph



Once this is created, simply right click, and navigate to Easy Dialogue->Easy Dialogue



Once created, you may click the “Character Dialogue” dropdown and type in any text that you would like the character to say.

3. Create an interface to EasyDialogueManager

This can most easily be referenced if looking at the script in the sample scene, however there are really only two things you need to call.

1. EasyDialogueManager.StartDialogueEncounter(ref EasyDialogueGraph _graph)
2. EasyDialogueManager.GetNextDialogue(ref EasyDialogueGraph _graph, out dialogue_line _outLine, ushort dialogueChoice = 0

All you need to be able to call these functions are, a reference to the graph that you want dialogue from, and a dialogue_line variable to hold the output dialogue_line data. Once you call these functions you should get back any information you care about inside the dialogue_line and you can display them or do whatever you please, however you want.

Support:

If there are any issues encountered, or questions, please feel free to reach me at chrismccole1@gmail.com. If there is enough demand I might try and start a discord server, if you feel that would be helpful to you, please let me know, as that would be valuable information.

Also, please let me know your honest feedback about EasyDialogue, and any features you would like added to the asset. Thank you!