

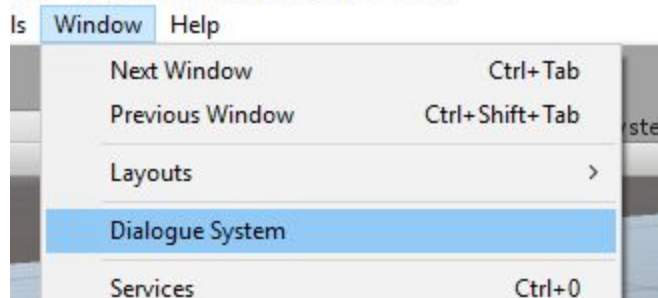
# Simple Dialogues

Hello! Thanks for looking at Simple Dialogues. This is a simple tool to create diverse dialogue trees that you can access via code to do whatever you want! This was made with programming in mind.

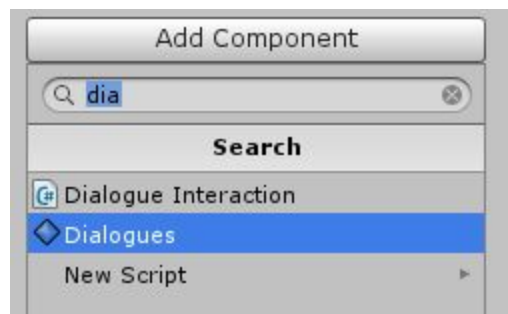
## Dialogue Editor

After downloading Simple Dialogues you can find the Dialogue Editor window by going to Window > Dialogue System.

Project - PC, Mac & Linux Standalone <DX11>



Once you open it, it will probably tell you that you have nothing selected. First, we must give a GameObject the dialogues component! Chose a gameobject to act as your NPC, go to add component, and search for Dialogues or drag and drop it from the scripts folder.



Now, when you have that GameObject selected and highlighted in the hierarchy, when you go to the Dialogue Editor it will show you a blank screen that allows you to add and remove trees!

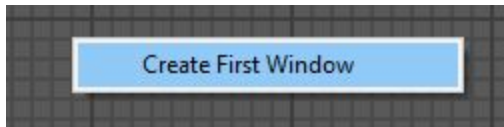
## Trees

Trees are basically our workspace, inside of this area we can add all of our nodes. You can have as many trees as you want. To add a tree, you'll just select "New Dialogue Tree" at the top.



Once you've done this you'll see our new workspace available! To move around in this area you'll want to hold down the middle mouse button and drag, this will allow you to scroll to all the bounds of the space.

Now you can add your first node by right clicking anywhere inside the grid space and hitting "Create First Window"



Once you've done that a window appears! From here you can do many things.

## Nodes

Nodes consist of a few elements. First the most obvious part, the big window in the middle, this section allows you to type your dialogue! There are also a -/+ button in the top left and right corners. These allow for quickly adding new window connections, or you can manually right click on the node, hit "Create Connection", then right click elsewhere and select either "Create Dialogue Window" or "Create Decision Window". Nodes also have triggers, but we'll get to that in a bit.

All nodes also should have a start and end, these are automatically generated. They mark where the nodes will reset to when using them in code, and when the current tree has ended.

Nodes have a few different types, they can either be a **Dialogue** window, a **Decision** window, or an **Option**. We'll go over each type.

## Dialogue Window

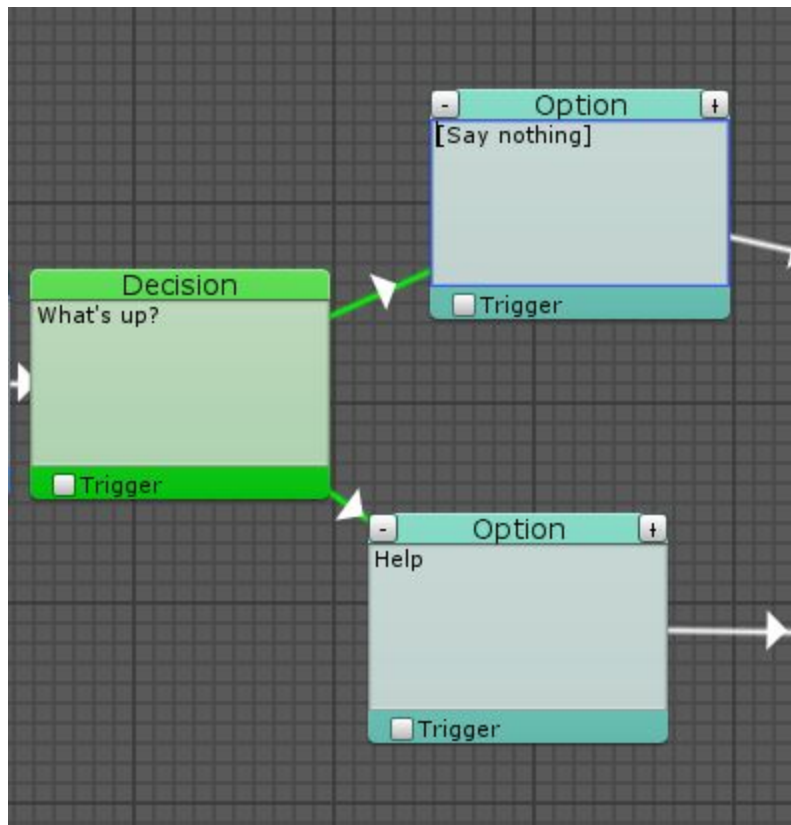
This is the most basic type of window, and it's pretty self explanatory. This is a window for just normal dialogue text to go into. It can only ever have one connection to another window.

## Decision Window

This window allows us to make decisions about where we want to go on the tree! If we want a window to have more than one branch, it has to be marked as a decision window. When you create a decision window, if it has no connections it will be yellow, indicating that it needs at least one, otherwise it will be green.

## Option Window

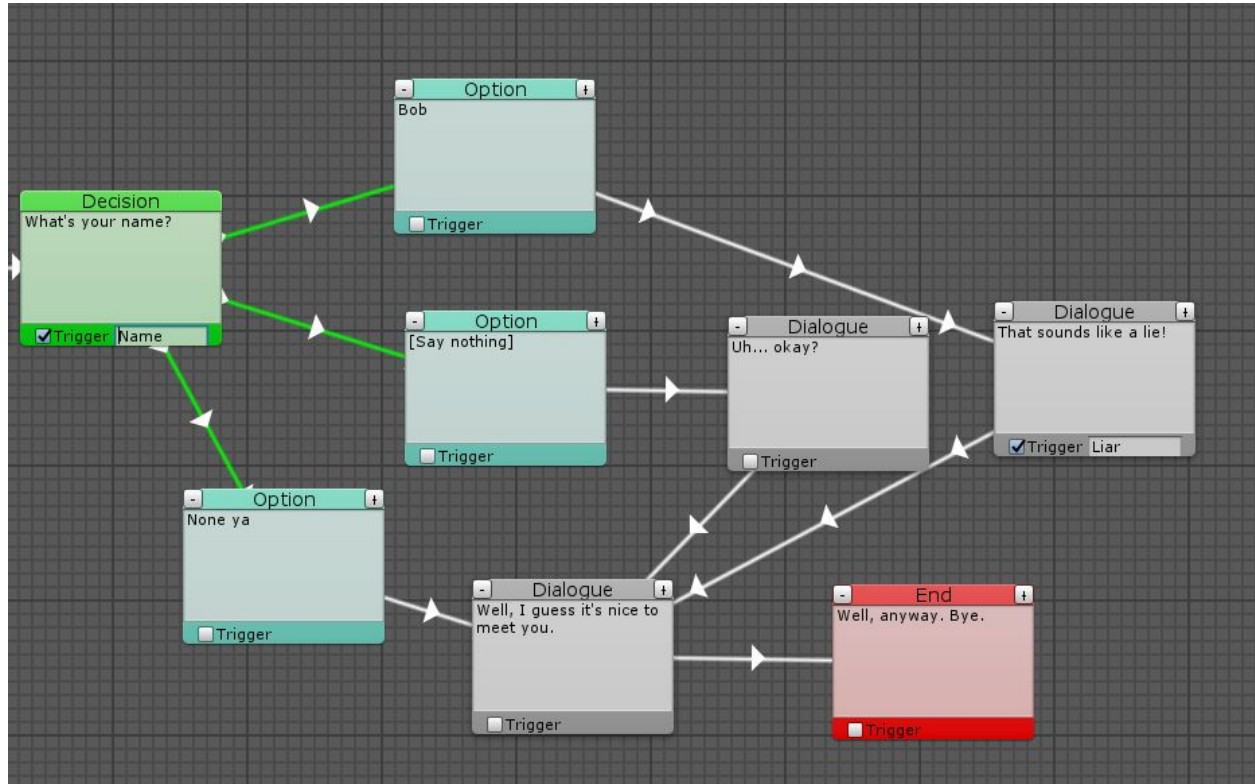
This type of window you cannot create manually, they are results from a decision window. When you create a decision window, any windows that are connected after that will be marked as option windows. These are to represent responses to whatever that decision window is, for example:



Here you see option windows are shown in cyan.

## Connections

Most of these nodes are created for you, but there are times you may want some windows to loop back together, or to revert to a previous node (such as a shop system). You can use the “Establish Connection” tool to do this. If you right click a node, hit “Create Connection” and then right click another window and hit “Establish Connection” those windows will now be connected! This doesn’t work with all windows, for example trying to make a connection between two nodes that already have a connection.



## Triggers

Triggers are a pretty simple feature, it just allows you to mark nodes as having the trigger attribute. If you hit the checkbox next to trigger on a node, a text box will appear. This allows you to type your trigger text that you can then access later in code.

# Dialogue Coding

Once we have our tree setup, then we can access all of the data from code! Included within the package is an example, you can look through the basic example there to get a good idea of how a basic dialogue system could be setup with this.

## Dialogues Component

The Dialogues component is the actual component you put on your NPC and what is used to gain access to all the dialogue you need. Once you gain access to this through a serializedfield or some other method, you can call the functions within it to get the data. We'll go through some of those functions here.

### String Reset()

This simple takes the current tree we're in, and resets the node we're tracking to the starting node.

### Bool SetTree()

This sets what tree we're working in, and where the dialogue should be pulled from. Calling this function automatically resets the tree.

### Bool End()

This simply returns whether the current node we're looking at is an end node.

### Bool HasTrigger()

Triggers whether this node has a trigger.

## String GetTrigger()

Returns the current trigger on this node.

## Int Next()

This moves our current node we're looking at to the next node in the tree. There are a few things to note however. If we're at the end of the tree, this will return -1. Also, if this node has choices to make, it will not progress and will return the number of choices it has, otherwise it returns 0.

## Bool NextChoice(string)

This function is what actually selects a choice based on the passed in string. It will evaluate the choices this node has, and if they match the string, it will go down that path, you can get the available choices with GetChoices(). Note, if this node is not a decision node, it will return false.

## String[] GetChoices()

Returns an array of strings of all the choices this current node has stored.

## String GetCurrentDialogue()

This actually returns the current dialogue of the node we're looking at.

# Feedback

Thanks for looking into Simple Dialogues! I already have future plans to support and add to this asset. If you have any feature suggestions or bug reports to make, please email in detail to [KoseckCory@gmail.com](mailto:KoseckCory@gmail.com).