



Dialogue System Documentation

Digital Worlds Unity Starter Package by Logan Kemper - Version 1

[Dialogue Guide](#)

Project Assets	
 File	 Description
Dialogue Prefab	A prefab containing all the GameObjects needed to display dialogue on the UI.
DialogueManager.cs	DialogueManager handles the dialogue that is sent by the DialogueTrigger. The sent text is navigated by DialogueManager and then displayed on the UI.
DialogueTrigger.cs	Sends dialogue to the DialogueManager via trigger collision or button press.
SpeakerLibrary.cs	A scriptable object that is used to supply DialogueManager with dialogue speakers and their associated sprites.

Dialogue Prefab

- Dependencies: DialogueManager.cs, DialogueTrigger, TextMesh Pro

Drag the Dialogue Prefab into the hierarchy as a child of the canvas.

DialogueManager.cs

- Dependencies: DialogueTrigger, TextMesh Pro

The Dialogue Prefab should already have DialogueManager mostly ready to go. Assign a SpeakerLibrary to the DialogueManager, once it has been set up.

DialogueTrigger.cs

- Dependency: DialogueManager.cs

Attach to a GameObject with a trigger collider. Assign the DialogueManager and a text file with the dialogue to be delivered. On a trigger collision with the player, the dialogue will be sent to

the DialogueManager. If the triggerType is set to Key Press, the player must be in the trigger volume and press the E key.

SpeakerLibrary.cs

To create a SpeakerLibrary scriptable object, right-click in the project window and choose Create > Speaker Library. For each character with dialogue, create a new SpeakerInfo entry by clicking the plus button, then filling in a name, optional portrait sprite, and name and text colors. The scriptable object can then be assigned to the [DialogueManager](#).

Tips

- The dialogue system will work in both 2D and 3D games. DialogueTrigger checks for both 2D and 3D trigger collisions, and every other script works independently from 2D/3D systems.
- Use the onDialogueBegan and onDialogueEnded UnityEvents on the DialogueManager to control systems that may not work correctly while dialogue is being delivered. For example, call the EnableMovement() method on the player controller and set it to false onDialogueBegan. Then set it to true onDialogueEnded. This will prevent the player from receiving movement input during dialogue. Calling ResetMovement when movement is disabled will freeze any residual momentum.
- DialogueManager will enable and disable the Dialogue Parent GameObject as needed, so it can be disabled in the hierarchy as to not block the game view unnecessarily.

How to Create Dialogue for the Dialogue System

DialogueManager expects dialogue to be formatted in a particular way to correctly display on the UI. Each “conversation” requires a plain text file assigned to a DialogueTrigger.

- Create a new text file. An example text file called SampleDialogue.txt should be in the dialogue system folder, but any .txt file will work. Duplicate the text file and name it something that corresponds to the conversation (e.g. “TutorialMessage1”).
- Open the text file with a text editor (VisualStudio, Notepad, TextEdit, etc.).
- Separate each line of dialogue to a separate line in the file.
- Prefix each line with the following: [NAME=Name][SPEAKERSPRITE=Name]
- “Name” can be replaced with the name of the speaking character. This name should exactly correspond to one of the name fields in the SpeakerLibrary.
- [SPEAKERSPRITE=Name] can be omitted if there is not a speaker sprite associated with that character.
- Assign the text file to a DialogueTrigger.

Example conversation:

[NAME=Player] Hello!

[NAME=NPC][SPEAKERSPRITE=NPC] I like your hat.

[NAME=Player][SPEAKERSPRITE=Player] I do not like your hat.