Dialogue and Quests

How to use Dialogue and Quests for Unity

The best way to get started with the Dialogue System is to look at the demo scenes. The following document helps to explain more in details how to create dialogues and quests.

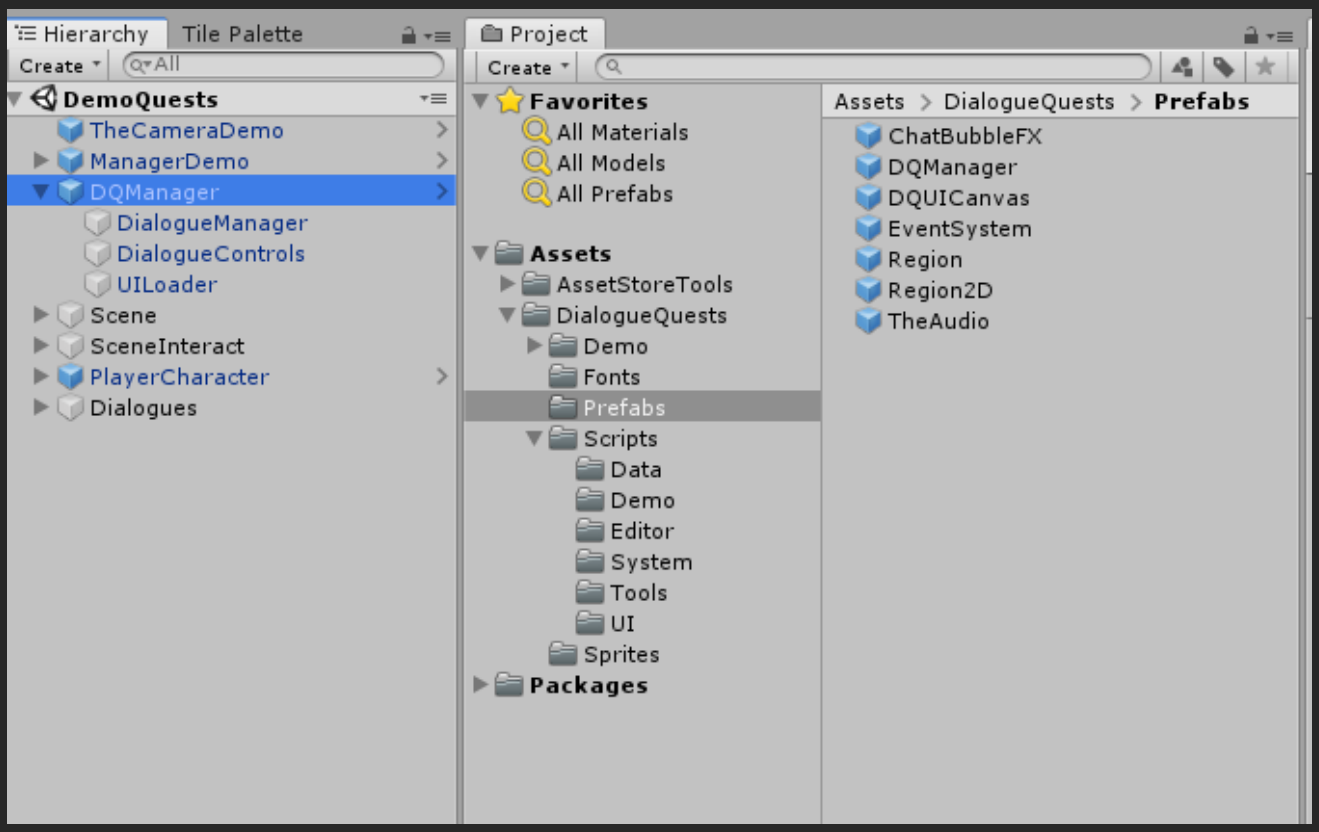
Getting Started

There are 3 things you will need to start creating dialogues:

- The Narrative Manager
- A Narrative Event with Dialogue Messages
- And an actor.

DQ Manager

Must be included once in your scene when you want to use the dialogue and quests system. It can be found in the prefabs folder.
This manager run the events and dialogues, loads the UI (DQUICanvas), and handles all user controls.

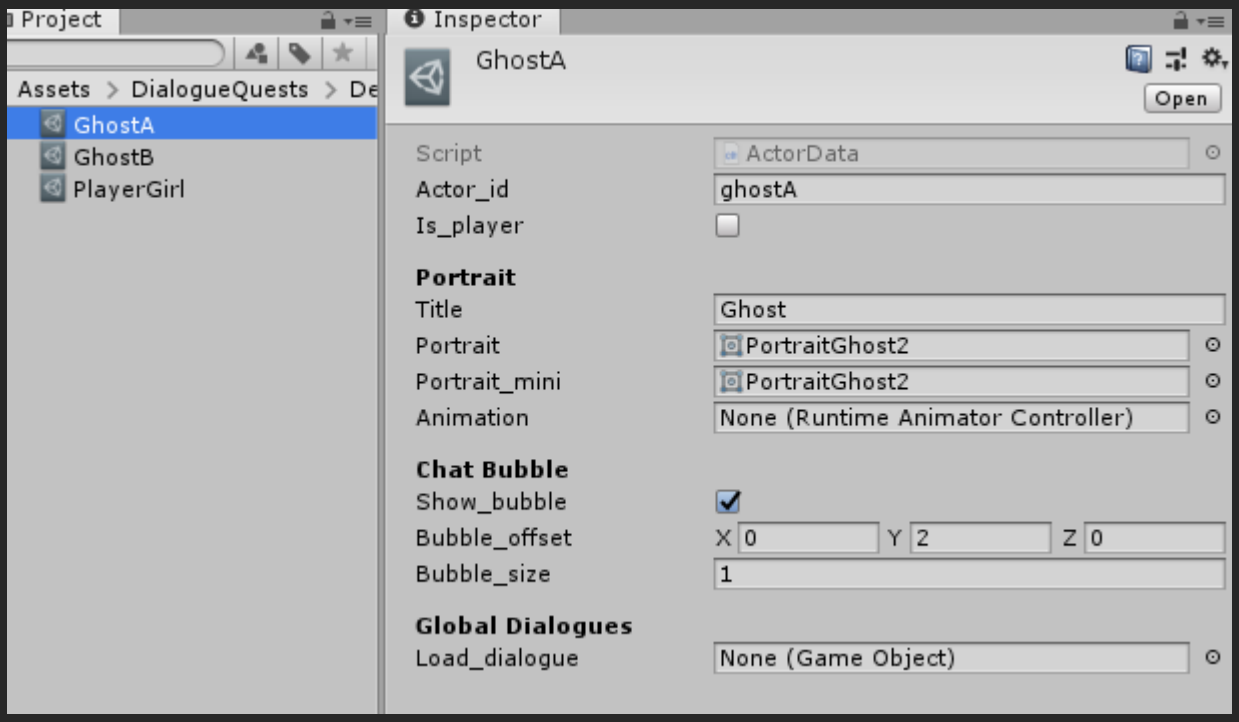


How to edit the UI?

You can do this by opening the DQUICanvas prefab contained in the Prefabs folder. To see an invisible panel, set the alpha value to 1 (on the CanvasGroup), this value is changed by script, but while editing its useful to set it to 1 just to see what you are working on.

Actors

Actors are character that are allowed to participate in a dialogue, they have a portrait, title and can trigger events or be set as the one saying a specific dialogue message. Make sure to add the Actor script on the Player character, and to check is_player, to allow this character to initiate dialogues.



Actor are defined in a scriptableObject (right click -> Create -> DialogueQuests -> Actor).
It's best to put the files in the Resources/Actors folder to have them loaded every scene (see the Loader script in the DQManager to set the path where they are loaded).

Actor_id: a reference unique to each actor, used in the save file.
Is_player: check for the player character.

Chat bubble: Just a visual effect that will show a speech bubble above the head of the actor when they are talking.

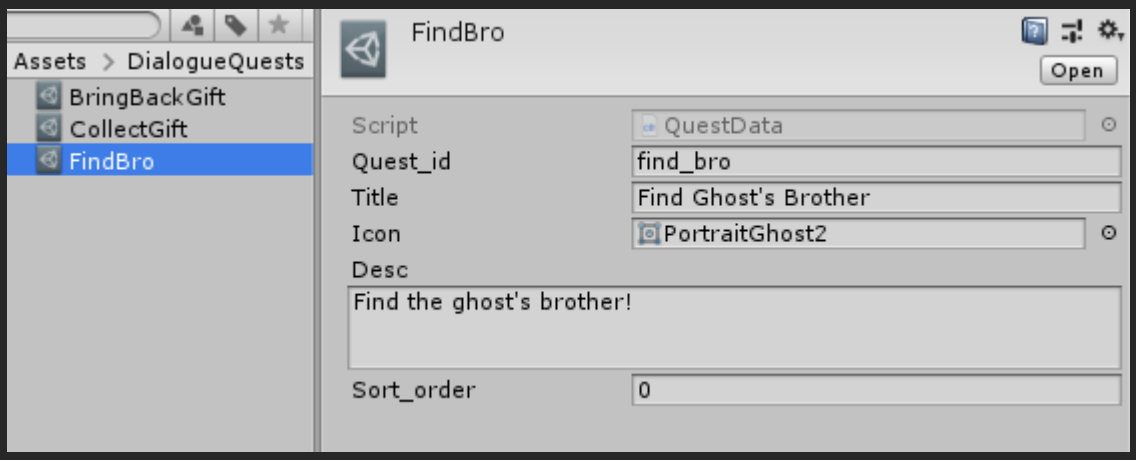
Animation: Its possible to add an animation to portrait of an actor. To test the animation while in edit mode, you should set the animator controller on the portrait in the UI (DQUICanvas, set the panel canvas_group alpha value to 1 to make it visible and test the animation).

Load_dialogues: Optional: any prefab set there will be spawned in every scene containing this actor. This allow to create events that are not scene dependent.

Once your actor data file is defined, you need to add the Actor.cs script to a prefab or object in the scene. And set the right actor data on it.

Quests

Quest are also scriptable objects (right click -> Create -> DialogueQuests -> Quest). And loaded from the Resources/Quests folder (by default).



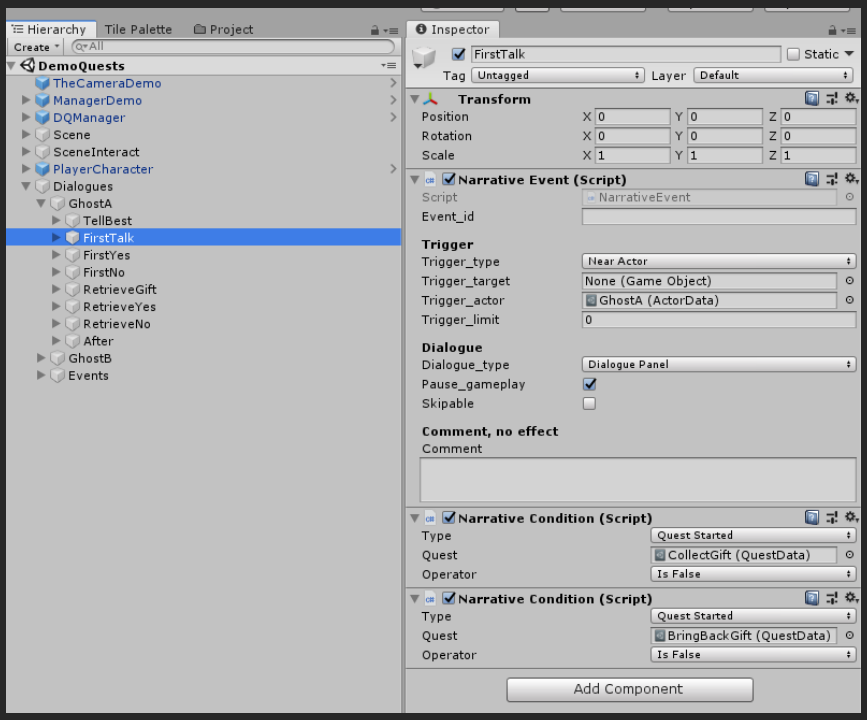
Actor_id: a reference unique to each quest, used in the save file.
Title and Desc: Text displayed as title and description of this quest.
Icon: Icon for the quest. (optional).
Sort_order: Optional, to sort the quest in a specific order in the quest panel.

How to trigger a quest?

Quest are started, and completed by using NarrativeEvents and NarrativeEffects (see next pages). Quest have 4 possible status: not started, active, completed, failed. You need to create events to start/complete/fail the quest.

Quest Journal?

Quests will appear in the Quest Panel (open it with J by default), once they are started or completed. Q



NarrativeEvent.cs

A dialogue event is a group of messages or actions that can be triggered in various ways. It can contain one or more dialogue messages, as well as one or more gameplay effects. You can define an event in your scene by creating an empty object and adding the NarrativeEvent.cs script to it.

Event_id is an optional ID reference to the event if you want its state (such as number of time triggered) saved to file.

Dialogue_Type Defines which dialogue panel will appear if the event contain dialogues.

Pause_gameplay will pause the game while this event is running. If you want this to work you need to link to the UnityAction onPause and onUnpause and pause your game. (NarrativeManager).

Skipable: Dialogue can be skipped (right click by default)

How are events triggered?

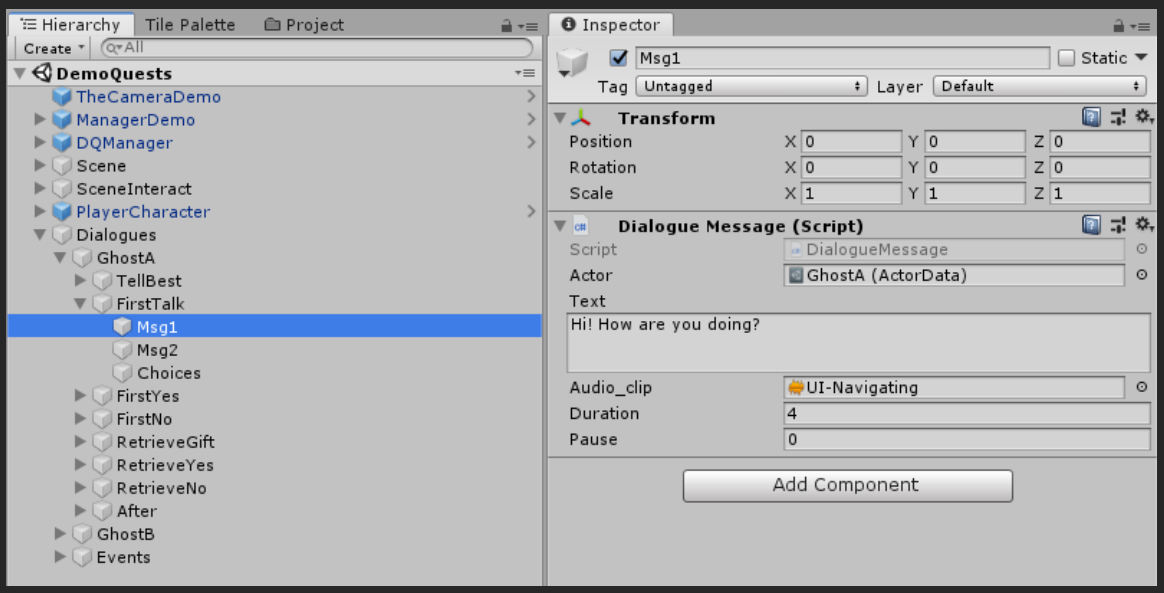
You choose how you want each event to be triggered, it can either be by interacting with an actor (InteractActor or NearActor), entering a region (EnterRegion), when loading the scene (AtStart), or you can set it to Manual and call it from the code with Trigger(). Regions are prefabs that you can add to the scene. (invisible collider that will trigger things when the player enter them).

Trigger_target: Reference to object that will trigger this event (ex: region)
Trigger_actor: Reference to the actor triggering this event (if set to ActorInteract/ ActorNear)

Trigger limit: will determine how many times that event can be triggered, with **0 meaning infinite**. If the event has an event_id the trigger count will be saved into the data file.

DialogueMessage.cs

Dialogue messages must be added as child objects of a NarrativeEvent, in the order you want them to appear.



Actor is the character that will tell that message. The character must have the DialogueActor script.

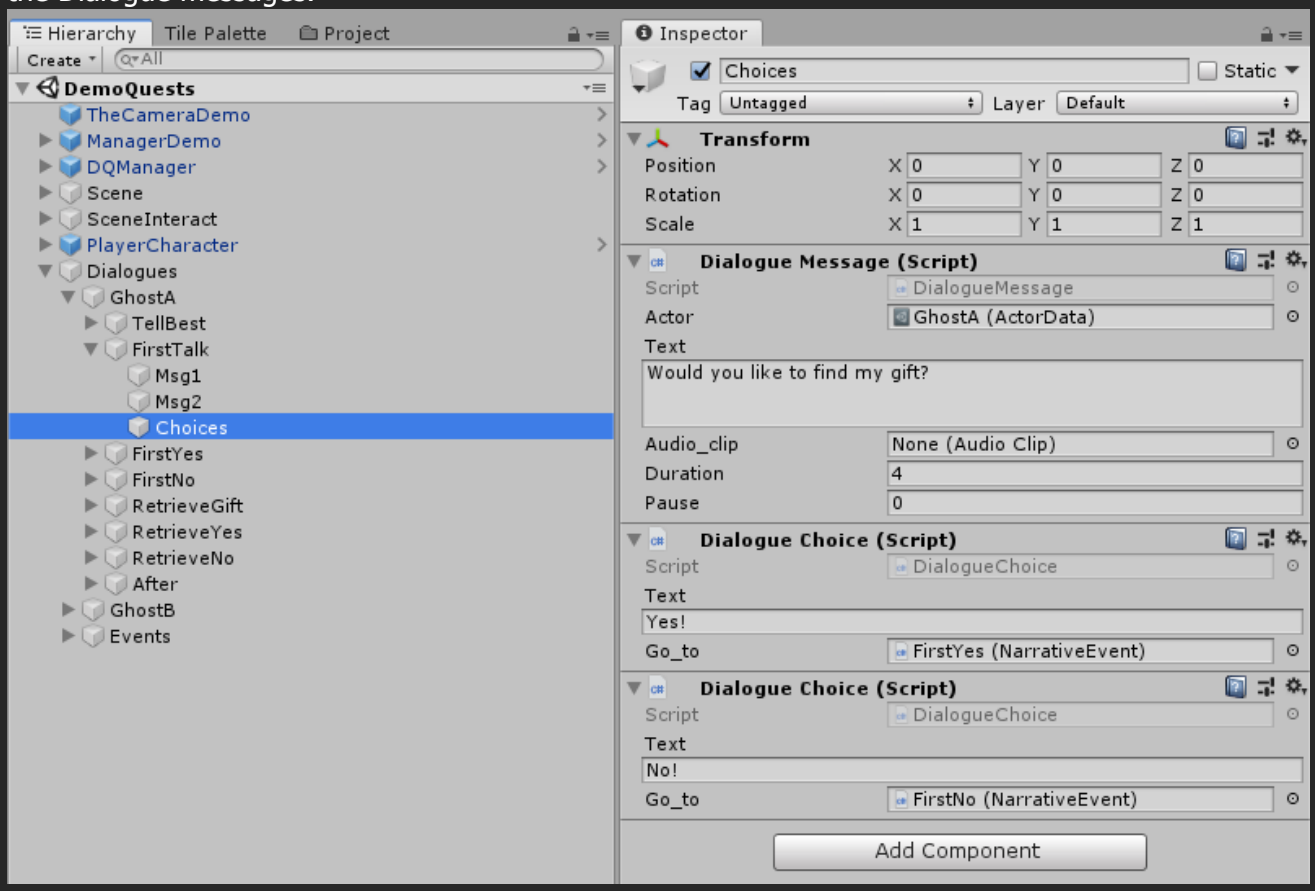
Audio clip: is a sound played during this message.

Duration: how long (in seconds) this dialogue will be shown (only if the dialogue_type selected is InGame, In DialoguePanel mode, the user must click to continue).

Pause: Pause time (in seconds) between this message and the next one.

DialogueChoice.cs

Dialogue chocies must be added as child objects of a NarrativeEvent, on the same object as one of the Dialogue messages.



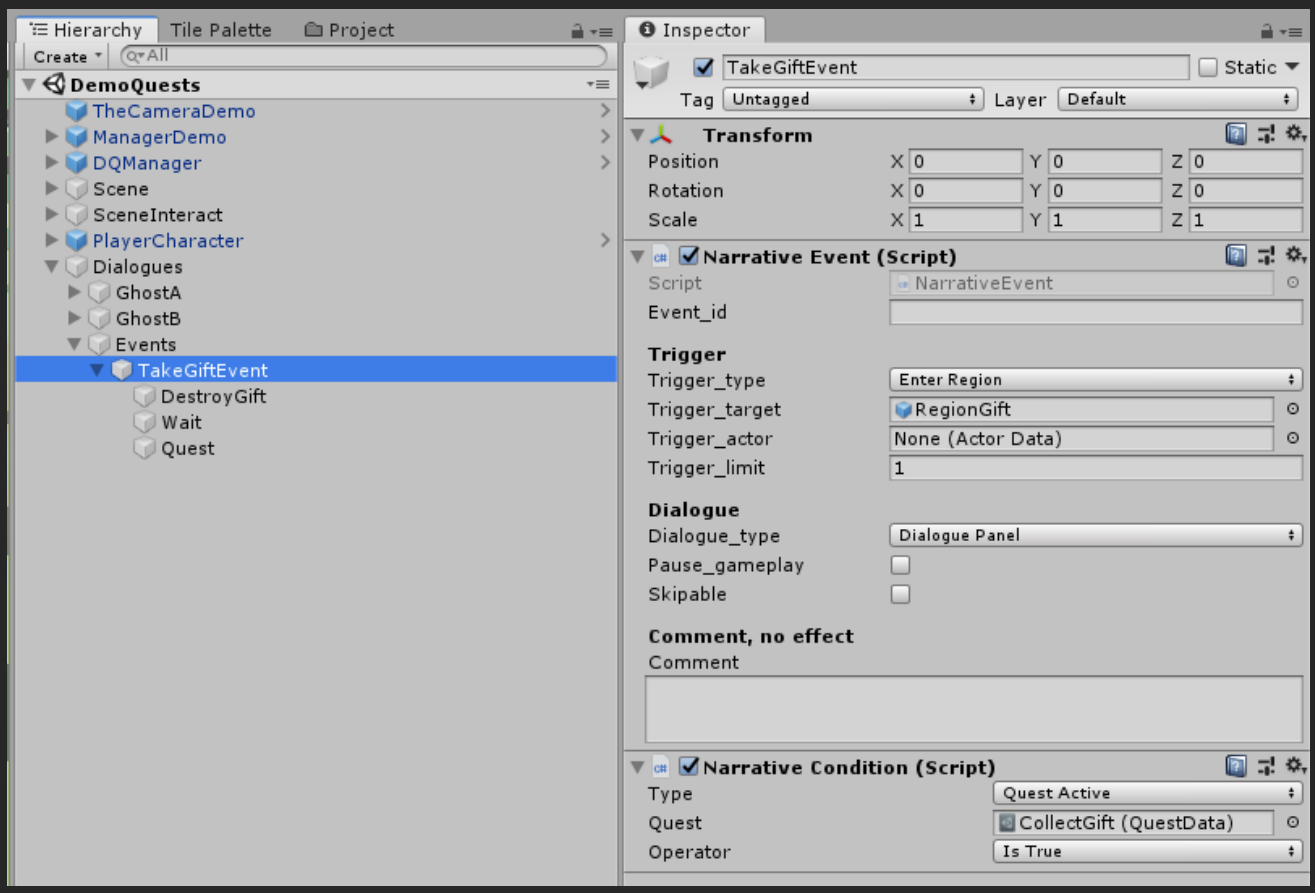
The choices will appear in the same dialogue box than that message. The player can then select one of them and it will trigger a new narrative event.

Go_to: Link to the narrative event that will be triggered when the player selects that choice. You can set that target event's trigger to Manual so it's not triggered by anything else than the choice.

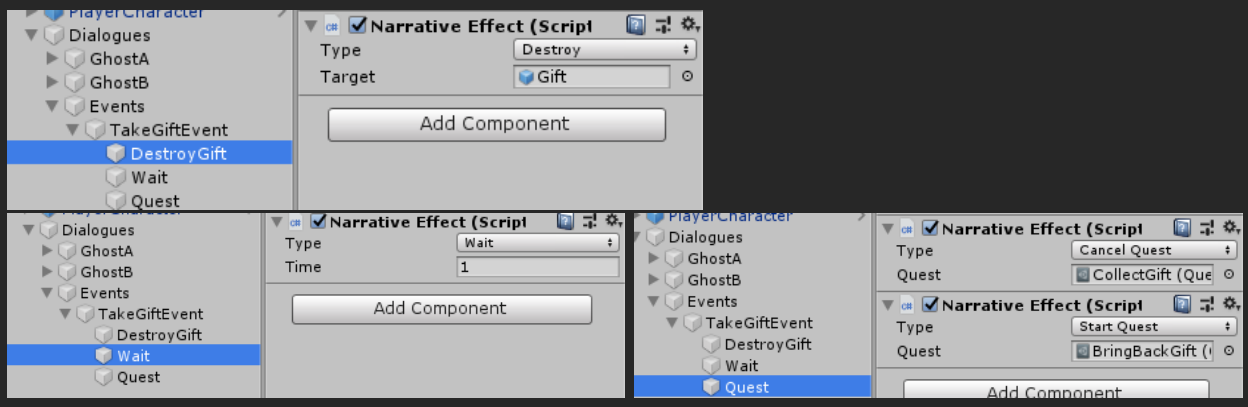
In the example above, when the FirstTalk narrative event is triggered, it will first play the Msg1 dialogue message, then Msg2, and the the 3rd message will contain 2 choices: Yes or No.
The first choice will trigger the FirstYes event, while the second will trigger the FirstNo event.

Conditions and Effects

Optionally, conditions and effects can be added to NarrativeEvents or to a Dialogue Messages. You may also have events that only contain effects (with no dialogues), in that case they must be set as child objects of a narrative event.



The TakeGiftEvent is triggered when the player enters the region RegionGift. It can only be triggered once and only if the CollectGift quest is active. (started but not completed)
When triggered, it will first destroy the gift object in the scene, it will then wait 1 second, and will finally end a quest and start a new one.



NarrativeEffect.cs

Effects are really just code functions that are called when an event is triggering. Child objects of an event are triggered one by one, so you can choose in which order effects will be executed if you have more than one line beneath the NarrativeEvent.

Effect Type:
Custom Int/Float/String allow you to save custom data for future conditions.
Show/Hide will SetActive or not an object
Spawn/Destroy will Instantiate or Destroy an object.
Start/Cancel/Complete/Fail Quest allow you to change the status of a quest.
ActorRelation: Allow you to add to, or set the value of an actor relation. (see it as a value that indicates the player progress with this actor, or how much this actor like the player) This doesn't do anything by itself, but you can use that value as a condition in other events.
Start Event: Allow you to start another event. (No conditions)
Start Event If Met: Allow you to start another event, but only if its conditions are true.
Play SFX/Music: Just play a sound or start a music.
Wait can add a delay before the next effect.
CustomEffect: Create a script that inherit from CustomEffect.cs, and add your own code to it within the DoEffect() function. Then generate a scriptableObject file from it, and use that custom effect in as many events as you want. (Will do a tutorial video on this).
CallFunction allow you to call any function of your choice.

NarrativeCondition.cs

Conditions define a requirement for an event or dialogue or effect to be triggered. For example, you could have an event trigger only when a quest is active, or only when an Actor's relation is higher than 10. Or when a custom value is set to a specific value.

If more than one condition on the same event, all of them must be true for the event to trigger. Conditions can also be added to dialogues messages to simply skip that message when the condition is false, or to effects to have conditional effects.

Condition Type:
Custom Int/Float/String will check if a custom value is set to a specific value.
IsVisible will check if a GameObject exists and if its active.
InsideRegion will check if a GameObject is inside a region.
Quest Started/Active will check quests status.
EventTriggered will check if another narrative event was triggered before.
ActorRelation: Check an actor's relation value set by other effects.
CustomCondition Create a script that inherit from CustomCondition.cs, and add your own code to it withint the IsMet() function. Then generate a scriptableObject file from it, and use that custom condition in as many events as you want. (Will do a tutorial video on this).

Improving the System

If you notice a missing feature, and you think that it would be really helpful for you. Please let me know about it. I really want to improve this system and make it great! And since I can't predict all the use cases, your feedback would really help me know what I should include in the future versions.

You can visit my youtube channel for tutorials:
https://www.youtube.com/channel/UC0LYig0AgPT9T5IN5DCUiqQ

If you have any questions or suggestions send me an email:
contact@indiemarc.com

Thank you!

Credits

Indie Marc (Marc-Antoine Desbiens)
Freelance Game Developer
indiemarc.com