Dialogue System for Unity
Visual Novel Framework

Version 2.0

This package contains a visual novel framework for the Dialogue System for Unity. You must also import the Dialogue System for Unity (https://www.assetstore.unity3d.com/en/#!/content/11672) version 2.0 or higher and use Unity 2019.4 or higher.

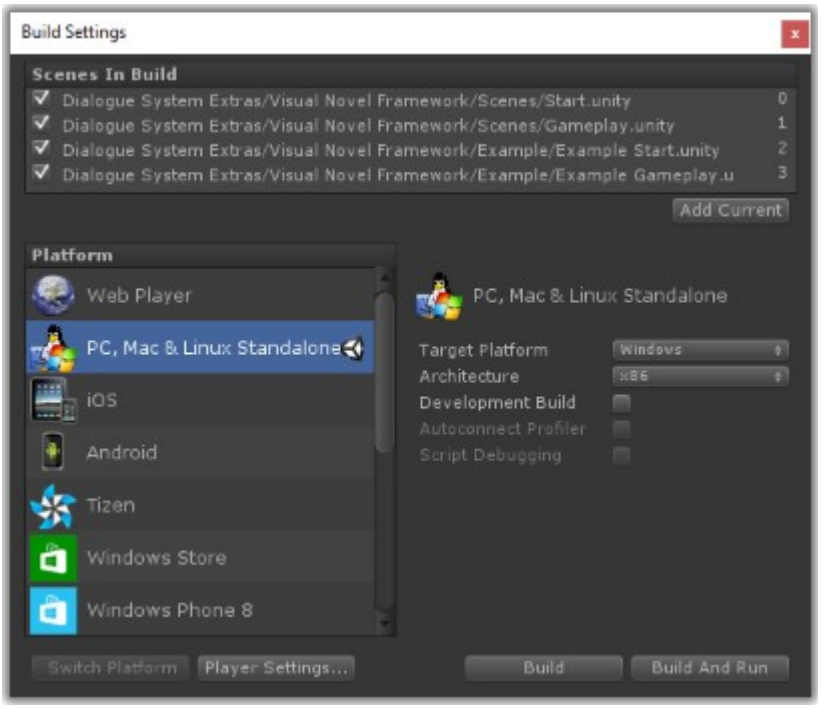
This framework assumes you have a basic familiarity with the Dialogue System. To familiarize yourself with the Dialogue System, click here.

Example Game

To play the example game, you must add the two scenes in the Example folder to your project's Build Settings:

1. Select the menu item **File > Build Settings…**

2. Drag both scenes (*Example Start* and *Example Gameplay*) into the **Scenes to Build** section.



Then play *Example Start*.

The example is very simple, but it will give you an idea of how the framework plays. Although the example is simple on purpose to make it easy to pick apart and understand, the framework itself is quite powerful and flexible.

How to Make Your Own Visual Novel

Follow the steps below to make your own visual novel:

Initial Setup

1. Select the menu item **File > Build Settings…**

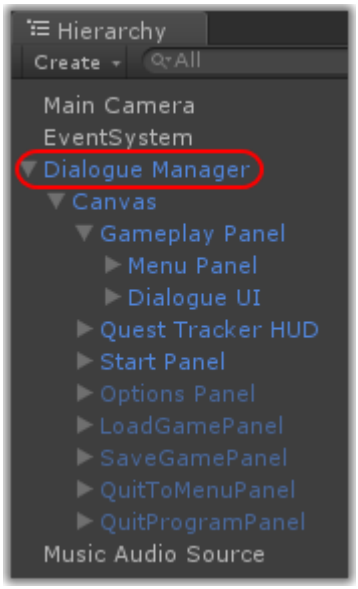
2. Drag all three scenes in the Scenes folder (*Start*, *Credits*, and *Gameplay*) into the **Scenes to Build** section as scene numbers 0, 1, and 2. If this section still has the example scenes, delete them or move them to the bottom of the list.

3. If you're using Ink:
   - Select **Edit > Project Settings > Player**, and add Scripting Define Symbol: `USE_INK`
   - Open the Start scene.
   - Add a Dialogue System Ink Integration component to the Dialogue Manager. Assign your Ink story.
   - Inspect Visual Novel Menu Canvas' Save Helper. Click the circle next to Conversation to switch from dropdown mode to text input mode, and enter the name of your Ink story.

Start Scene

4. Open the *Start* scene.

5. The Start scene has a *Dialogue Manager* game object.



This is the engine that drives the Dialogue System. The *Dialogue Manager* game object will not be destroyed when changing scenes. It contains all of the user interfaces (UIs) and settings used by the Dialogue System and this visual novel framework.

In this framework, the content of a visual novel is contained in one or more “conversations,” which are lines of text linked together in a certain order.

a) The conversations are contained in a *dialogue database*. The *Dialogue Manager* already points to a starter dialogue database located in the Data folder. To edit it, click on the *Dialogue Manager*'s blue and white logo image.
To read more about editing dialogue databases, click here.
To use third party editors such as Chat Mapper or articy:draft, click here.

b) The *Dialogue Manager*'s properties specify how lines are shown, such as how long to show the text, whether to wait for a continue button click after each line, etc.
To read more about setting properties, click here.

c) In this framework, the *Dialogue Manager* has these modifications:

- Dialogue UI: Points to the template VN dialogue UI. For more info about UIs, click here.
- New component: Remember Current Dialogue Entry.
- New component: Handle Background Fields.

6. The Start scene also has a *Visual Novel Menu Canvas* that controls the main menu and the gameplay menu.

a) *Start Panel*: The main menu. As with all of the UIs, the menu graphics are placeholders. At some point you'll want replace the graphics and maybe even change the layout. The UI elements are also configured for language localization using the *Text Table* in the Data folder.
To read more about localizing UIs, click here.

You probably want to get to the meat of your game, so you can save this step for later.

Start Panel overlaps the other UIs. Temporarily deactivate *Start Panel* to see them.

b) *Gameplay Panel*: Contains gameplay control buttons such as Save, Load, Options, Menu, and Quit.

7. The *Start* scene also has a *Music Audio Source* game object. If you want background music in your main menu, set the *Music Audio Source*'s **AudioClip**.

Gameplay Scene

8. Open the *Gameplay* scene.

9. This scene contains a modified copy of the *Dialogue Manager* game object that lets you play your game directly from this scene instead of going through the *Start* scene first. This is just a convenience to be able to test your game more easily.

   If you modify the master copy of the *Dialogue Manager* game object in the *Start* scene, click the **Apply** button to apply the changes to the Unity prefab. The copy of the *Dialogue Manager* in the *Gameplay* scene will inherit those changes.

10. In the *Gameplay* scene, you can set a different background image, configure a different *Music Audio Source*, etc.

11. You can use multiple gameplay scenes if you want. To change scenes during a conversation, use the included VNLoadLevel() sequencer command, which has the same syntax as LoadLevel().

Writing Your Novel

To begin writing your novel, click on the *Dialogue Manager*'s blue and white logo. This will open the Dialogue Editor. Then follow these basic steps:

11. Click on the *Actors* tab.

    a) From the dropdown menu in the upper right, select **New Actor** to add each character.

    b) You can set one or more portrait images for each character. If you want to use animated portraits, click here or watch this tutorial.

    c) Optionally define a custom field named *Background* that specifies the background image to use when this actor is talking. Store the image inside a Resources folder in your project, and specify its name in the *Background* field. To add this custom field, expand the actor's **All Fields** section and click the "+" button. This will add a new, blank field to the actor. Then change the field's **Title** to *Background*.

12. Click on the *Conversations* tab.

    a) The dropdown in the upper left starts with one conversation titled *Start Conversation*. The framework will start this conversation when starting a new game. You can write your entire novel in this conversation, or you can define additional conversations and link between them. To create a link from a dialogue entry node to another conversation, select the node. In the Inspector's **Link To:** dropdown, select *(Another Conversation)*.

    Optionally define a custom field named *Background* to use when the conversation starts.

    Optionally define a custom field named *CurrentStage*. When the conversation starts, if the field exist and isn't blank, it will set the variable named *CurrentStage* to this value. When you save a game, the *CurrentStage* variable's value is shown in the summary text for the saved game slot. If you're localizing to multiple languages, you can create localized versions of *CurrentStage*, such as a *CurrentStage ES* field for for Spanish, a *CurrentStage FR* field for French, etc.

    b) Each dialogue entry node has:

        1. **Dialogue Text**: Text shown onscreen (spoken by actor).

        2. **Menu Text**: Optional form of text (usually shorter) shown in player response menus.

        3. **Sequence**: Plays sequencer commands to make things happen in the game. Sequences are incredibly useful to show and hide game objects such as images, play sounds, load new levels, and more. We recommend you create all of your additional images as game objects (such as UI images under the Canvas), and then use the SetActive() sequencer command to activate and deactivate them as needed.

To read about sequences, click here.
To read about language localization, click here.

- To record activated GameObjects in saved games, follow these steps:
  a) In the Sequence field, use SetActive() to activate the GameObject, and add Delay({{end}}) to keep the line onscreen for an appropriate amount of time. For example, you could use a sequence like this:
     ```
     SetActive(Girl Angry Image);
     Delay({{end}})
     ```
     This activates the GameObject named "Girl Angry Image" and delays for a duration based on the length of the line.
  b) On an active GameObject, add an Active Saver component, and point its Target field to the GameObject that gets activated (e.g., Girl Angry Image).

4. **Background**: Optional custom field (meaning you need to add it manually in the **All Fields** section) containing the name of an image in a Resources folder in your project. If this field exists and isn't blank, the framework will load the image and show it as the background image.

5. **CurrentStage**: Optional custom field to specify summary to record with saved games.

13. On the *Variables* tab, you can define variables to remember choices that the player has made.

    The dialogue database already has a definition for a variable named *CurrentStage*. You can set this variable at any point in the conversation. The contents of *CurrentStage* will be included in the summary shown next to each saved game in the load game menu.

    To read more about variables, click here.

14. On the Quests/Items tab, you can define quests. You can also use quests to show a "to do" list for the player.

15. Normally the game ends when the player reaches the end of the conversation. If your game incorporates other kinds of gameplay between conversations, inspect the Dialogue Manager's Menus component and *un*tick **Return To Menu On Conversation End**. To return to the main menu in this case, you will have to manually use the VNMainMenu() sequencer command.

Final Steps

15. Open the Credits scene. The game switches to this scene when the player clicks the *Credits* button in the main menu. Customize this scene to contain your own credits. The *Credits Panel* in the *Start* scene's canvas is shown on top of the scene. It contains a *Back* button that returns to the main menu. The *Back* button is currently configured to be invisible and cover the whole screen; this way the player can click anywhere to return to the *Start* scene.

16. When you're ready to distribute your game, open the Build Settings window.

17. Make sure you don't have any extra scenes in the **Scenes to Build** list such as the example game scenes. Then set your platform and click the **Build** button.

Quest Log

You can add quests as normal in the Dialogue System.

- The quest log window is instanted from a prefab using an Instantiate Prefabs component on the Visual Novel Menu Canvas.
- The gameplay buttons panel has a Quest Log button. If you don’t want to use a quest log, simply deactivate this button.
- The Dialogue Manager also instantiates a quest tracker HUD if you want to show quest tracking info during gameplay.

Achievements

The Visual Novel Menu Canvas has an Achievements component in which you can define achievements.

The canvas has an AchievementsPanel child panel. You can customize its grid layout and template.

To unlock an achievement, use the Lua function UnlockAchievement(“name”). For example, the example scene has an achievement named “Creep”. A dialogue entry in the conversation unlocks it with this Lua function in its Script:

```
UnlockAchievement("Creep")
```

Achievements are stored in PlayerPrefs, separately from individual saved games.

If you don’t want to use achievements, simply deactivate the Achievements button the gameplay buttons panel.

Background Images

Ways to Set the Background Image

You can change the background image using any of these techniques:

- Add a custom field named Background to a dialogue entry. Set it to the name of a background image located in a Resources folder. When this entry is shown, it will change the background image.
- Or add a custom field named Background to an actor. When the actor speaks a line, it will change the background image. (Dialogue entry Background fields take precedence.)
- Or use the Background(*filename*) sequencer command. This will change the background to the specified image in a Resources folder.

Background Transitions

You can use the Dialogue Manager's Scene Transition Manager to fade to black before changing backgrounds and then fade back in after changing backgrounds. To do this, set the Background Manager component's **Fade Duration** field or use the Lua function BackgroundFadeDuration:

```
BackgroundFadeDuration(3)
```

This Lua function will set Fade Duration to 3, which will fade to black, wait 3 seconds, and then fade back in. If Fade Duration is zero (the default), there will be no fade when changing backgrounds.

Need Help?

We're here to help! If you have any questions or feature requests, please contact us at:

Email: support@pixelcrushers.com

Forum: https://www.pixelcrushers.com/phpbb/index.php

Games Showcase

Write a game with the Dialogue System's Visual Novel Framework? We'll be happy to feature it on the Dialogue System's Games Showcase. Just send screenshots and/or a video and a link to the game's website to support@pixelcrushers.com!