



DIALOGUE SYSTEM

for unity

Addon for OpenAI

Copyright © Pixel Crushers

Table of Contents

Introduction.....	2
We're Here to Help.....	2
Quick Start.....	2
OpenAI and the Dialogue System for Unity.....	2
Language Models.....	3
ElevenLabs Speech Synthesis.....	3
Overtone Speech Synthesis.....	4
Dialogue Smith.....	4
General AI Assistant.....	4
Welcome Window.....	5
Main Window.....	6
Generate Conversation.....	7
Revise Text.....	9
Translate Field.....	11
Translate Database.....	12
Freeform Chat.....	13
AI Assistant Window.....	15
Generate Portraits.....	16
ElevenLabs Voiceover Audio Generation.....	18
Overtone Integration.....	21
Dialogue Smith Integration.....	24
Runtime Dialogue.....	27
Questions? Feature Suggestions?.....	31

Introduction

Thank you for supporting the Dialogue System for Unity and the Addon for OpenAI! This addon requires the [Dialogue System for Unity](#). Using the Dialogue System Addon for OpenAI, you can use OpenAI's generative AI to write narrative content faster, get suggestions when you're stuck, fix spelling and grammar, and translate your text to other languages. You can also use the addon for freeform chat similar to ChatGPT, generate portraits using DALL-E, and voiceover using ElevenLabs.

We want your project to be a success! If you have any questions or feature requests, please contact us any time at support@pixelcrushers.com.

We're Here to Help

- Email: support@pixelcrushers.com
- Forum: <https://pixelcrushers.com/phpbb>
- Discord: <https://discord.gg/FwUaCNt>
- Web: Tools → Pixel Crushers → Dialogue System → Help → Report a Bug

Quick Start

- Read the documentation and/or watch the tutorial videos.
- Import the Dialogue System for Unity into your project.
- Import the Dialogue System Addon for OpenAI into your project.
- The addon's Welcome window will appear. Configure the addon to proceed.
- Click the OpenAI Addon Window button in the Welcome window or the AI buttons in the Dialogue System's Dialogue Editor window.

OpenAI and the Dialogue System for Unity

OpenAI is an AI research company best known for technology designed to generate human-like language, in particular using a language model called Generative Pre-trained Transformer, or GPT. OpenAI has a public API that applications such as the Dialogue System Addon for OpenAI can use to leverage the power of its language models. Modest charges for the use of the API are handled through an API key that you can create on OpenAI's website. ([OpenAI Pricing](#)) The Dialogue System Addon's Welcome window will step you through the process if you don't already have an API key.

The **Dialogue System Addon for OpenAI** is a third party product (i.e., not affiliated with OpenAI) that allows you to leverage the OpenAI API to enhance your content creation. It's designed to work seamlessly with the Dialogue System for Unity.

Note: Language models are not sentient. They are essentially just very advanced pattern matchers. Results may contain grammatical or factual errors, so review them carefully before using them.

Language Models

The OpenAI API provides access to several language models and parameters. In the Dialogue System Addon, the UIs to specify language models and parameters look like this:

A screenshot of a dark-themed user interface for configuring an OpenAI API. It features three settings: 'Model' is a dropdown menu currently showing 'GPT3_5_Turbo'; 'Temperature' is a slider with a circular knob positioned towards the left, and a numerical display showing '0.2'; 'Max Tokens' is another slider with a circular knob positioned towards the left, and a numerical display showing '1024'.

Model: This dropdown allows you to choose a language model for content generation:

Model	Description
GPT_3_5_Turbo	This and GPT-4 Turbo are best for most cases.
GPT_3_5_Turbo_16K	Same as GPT-3.5 Turbo but with 4x the context window for longer responses.
GPT_4	Newest, most powerful language model. Improves on GPT-3.5.
GPT_4_32K	GPT-4 with larger 32k context window that allows for very long responses.
GPT_4_Turbo	GPT-4 Turbo with 128k context window.
Davinci	Powerful model but slow. Good at complex intent and cause & effect.
Curie:	Second most powerful model. Good at language translation.
Babbage	Second fastest model. Good at classification.
Ada	Smallest, fastest, and least expensive model, but results may be poor.

Temperature: Specifies the amount of “randomness” to include in the content. A value of zero generates plain, straightforward responses. A value of one generates more unexpected or “creative” responses.

Max Tokens: Specifies the maximum number of tokens to spend on a response. Lower numbers generate shorter responses; higher numbers can generate longer responses. OpenAI pricing is based on the number of tokens used, so you can use this slider to reduce expenses.

ElevenLabs Speech Synthesis

You can use the API provided by ElevenLabs, a separate company from OpenAI that specializes in speech synthesis, to generate voiceover audio for your dialogue.

ElevenLabs uses a separate API key. Any charges for using the API are applied to your API key account. (See [Pricing](#).)

You can use the addon’s ElevenLabs features to generate audio at design time and at runtime.

Overtone Speech Synthesis

You can also use Least Squares' [Overtone](#) for text to speech. See [Overtone Integration](#).

Dialogue Smith

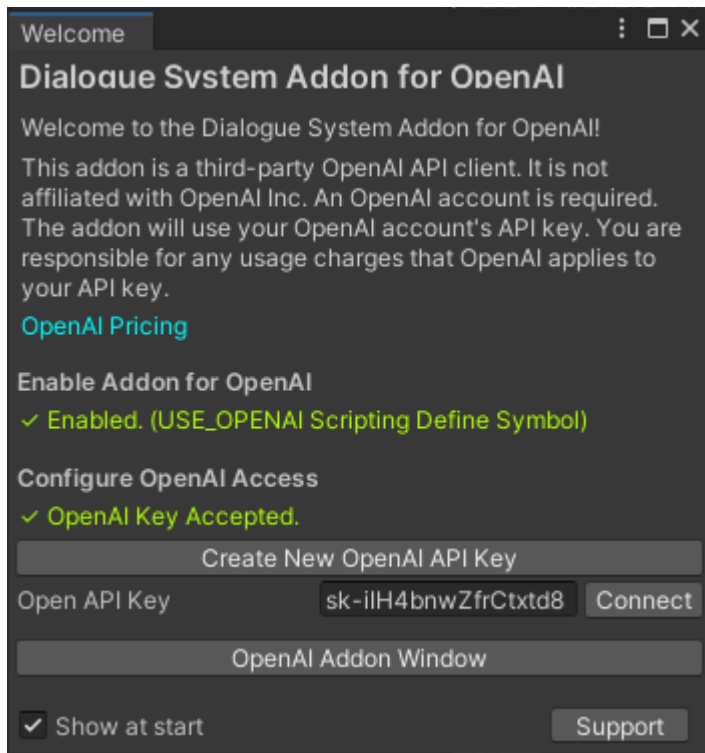
You can also use the API provided by Dialogue Smith, a separate company from OpenAI that specializes in advanced dialogue generation, to fill in dialogue for branching conversation trees.

Dialogue Smith uses a separate API key. Any charges for using the API are applied to your API key account.

General AI Assistant

The Dialogue System Addon for OpenAI also has a general AI Assistant window that's useful to ask questions in a ChatGPT-like interface.

Welcome Window

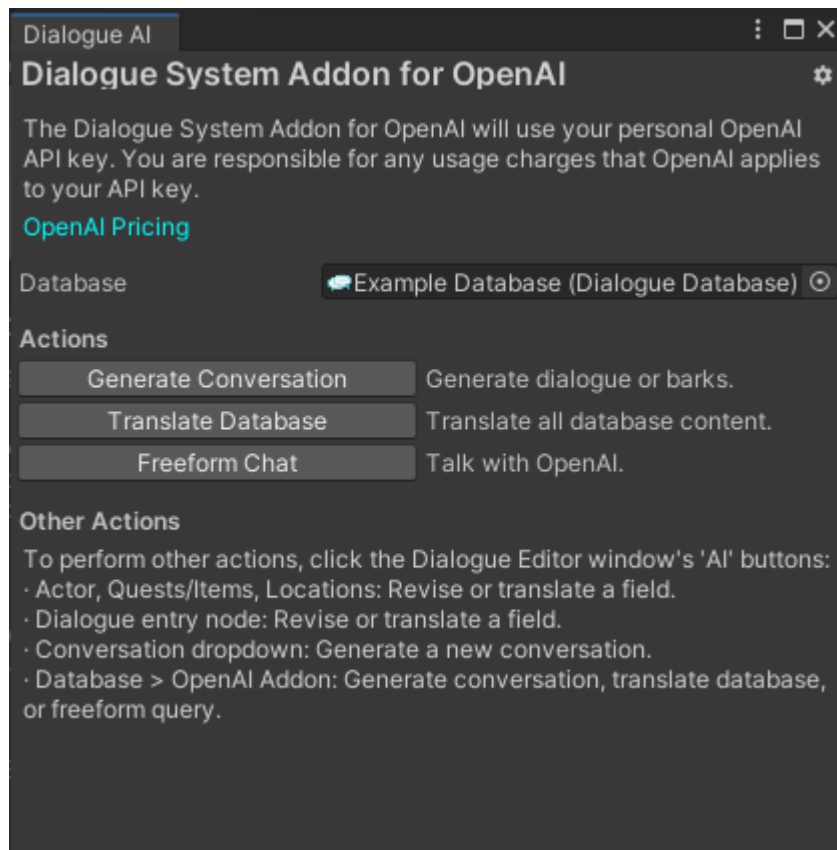


Use menu item **Tools** → **Pixel Crushers** → **Dialogue System** → **Addon for OpenAI** → **Welcome Window** to open the Welcome window.

The Addon for OpenAI will use your OpenAI API key, which you can create on OpenAI's website. You are responsible for any usage charges that OpenAI applies to your API key.

You can use the Welcome window to create and enter an OpenAI API key, and to open the Main Window.

Main Window



You can access the Main window from the addon's Welcome window or menu item **Tools → Pixel Crushers → Dialogue System → Addon for OpenAI → Main Window**.

On the Main Window, you can access these windows:

- **Generate Conversation:** Generate a new conversation or set of barks.
- **Translate Database:** Translate your database to one or more languages.
- **Freeform Chat:** Talk with OpenAI in a format similar to ChatGPT.

The Addon for OpenAI also adds **AI** buttons to several locations in the Dialogue System's Dialogue Editor window. You can click these **AI** buttons to access additional features.

Generate Conversation

The screenshot shows a software window titled 'Dialogue AI' with a subtitle 'Dialogue System Addon for OpenAI'. It features two tabs: 'Main' and 'Generate Conversation', with the latter being active. Below the tabs, there's a description: 'Generate a conversation between two actors or a set of barks.' Two buttons, 'Generate Conversation' and 'Generate Barks', are present, with the first one highlighted. The form includes several input fields and sliders: 'Model' is set to 'GPT3_5_Turbo'; 'Temperature' is a slider set to 0.2; 'Max Tokens' is a slider set to 1024; 'Conversation Title' is 'War'; 'Actor' is 'Player'; 'Conversant' is 'Private Hart'; 'Extra Actors' has a '+' button; 'Action' is 'regarding'; and 'Topic' is 'the purpose of war.'. A 'Generate' button is located below the topic field. At the bottom, there's a 'Refinement Instructions' field with a 'Refine' button, and two large buttons: 'Accept' and 'Cancel'.

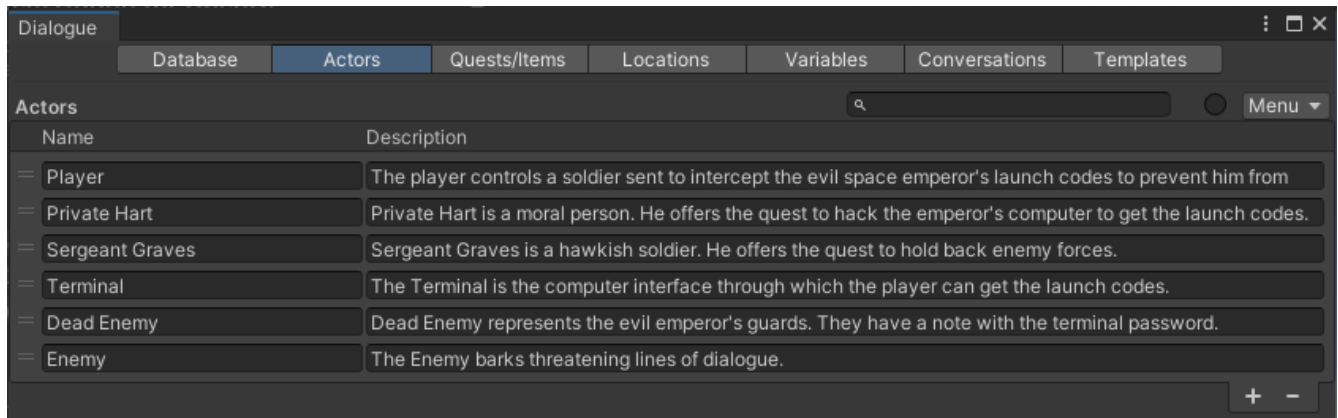
Access the Generate Conversation window from the Main Window. Here, you can create a new linear conversation between two actors or create a bark conversation.

Generate Conversation

To create a conversation, specify a title, participants, action, and topic. For example, the screenshot above will prompt the GPT_3_5_Turbo language model to generate a conversation titled “War” between Private Hart and Sergeant Graves regarding the purpose of war. Then click **Generate**.

When the Addon receives a response from OpenAI, it will show the generated conversation. If you want to adjust the conversation, enter Refinement Instructions and click **Refine**. To add the conversation to your dialogue database, click **Accept**.

When creating a conversation, the prompt will also include the participants' **Description** fields and any actors mentioned in the topic. Include the actor's name in the Description field, such as:



Knowledge & Goals Fields

You can optionally add fields named **Knowledge** and **Goals** to actors to contain additional information to provide in the prompt. To add these fields to all actors, open the Dialogue Editor's **Templates** section > **Actors Template** and click **Add Knowledge and Goals Fields**.

Location Descriptions

If you tick the Generate Conversation window's **Use Location Descriptions** checkbox (not shown in screenshot above) the prompt will also include the **Description** fields of all Locations in your dialogue database.

Branching Dialogue

To generate branching dialogue, you can use the addon's [Dialogue Smith integration](#).

Generate Barks

To generate barks, click the Generate Barks toolbar. The UI will appear the same as Generate Conversation with the addition of a Num Barks slider. Set the slider to the number of barks you'd like to generate, and click **Generate Barks**.

When you're satisfied with the results, click **Accept** to add a conversation containing the barks.

Revise Text

The screenshot shows the 'Dialogue System Addon for OpenAI' window with the 'Revise Text' tab selected. The window has a title bar with standard OS controls. Below the title bar, there are tabs for 'Main' and 'Revise Text'. The 'Revise Text' section includes a text input field with the placeholder 'Revise the text of a field.'. Below this are sliders for 'Model' (set to 'GPT3_5_Turbo'), 'Temperature' (set to 0.2), and 'Max Tokens' (set to 1024). A 'Dialogue Entry: Dialogue Text' section contains a text area with the text 'We need to intercept the launch codes before the enemy launches their weapon at our home world.'. Below this is a 'Num Variations' slider set to 3. A 'Refinement Instructions' section has a text input field with 'Make it more urgent.' and a 'Refine' button. A 'Fix Grammar & Spelling' button is also present. The 'Proposed Revisions:' section lists three suggestions, each with an 'Accept' button. At the bottom are 'Use All' and 'Cancel' buttons.

Dialogue AI

Dialogue System Addon for OpenAI

Main Revise Text

Revise Text

Revise the text of a field.

Model GPT3_5_Turbo

Temperature 0.2

Max Tokens 1024

Dialogue Entry: Dialogue Text

We need to intercept the launch codes before the enemy launches their weapon at our home world.

Num Variations 3

Refinement Instructions: Make it more urgent. Refine

Fix Grammar & Spelling

Proposed Revisions:

"It's a matter of life and death! We must intercept the launch codes before the enemy unleashes their weapon on our home world!" Accept

"The clock is ticking! We have to intercept the launch codes before the enemy fires their weapon at our home world!" Accept

"Our survival depends on it! We need to intercept the launch codes before the enemy attacks our home world with their weapon!" Accept

Use All Cancel

Access the Revise Text window by clicking the **AI** button next to a field in the Dialogue Editor such as the Dialogue Text field:

The screenshot shows the 'Dialogue Entry' window. It has fields for 'ID' (1), 'Title' (New Dialogue Entry), and 'Description' (Used if the quest is unassigned. Cuts to an immediate closeup of the speaker (Hart)). Below these are dropdown menus for 'Actor' (Private Hart [2]), 'Conversant' (Player [1]), and 'Group'. At the bottom, there are two text input fields: 'Menu Text' and 'Dialogue Text (95 chars)'. Each field has an 'AI' button to its right. The 'AI' button next to the 'Dialogue Text' field is highlighted with a red box.

Dialogue Entry

ID 1

Title New Dialogue Entry

Description Used if the quest is unassigned. Cuts to an immediate closeup of the speaker (Hart).

Actor Private Hart [2]

Conversant Player [1]

Group

Menu Text AI

Dialogue Text (95 chars) AI

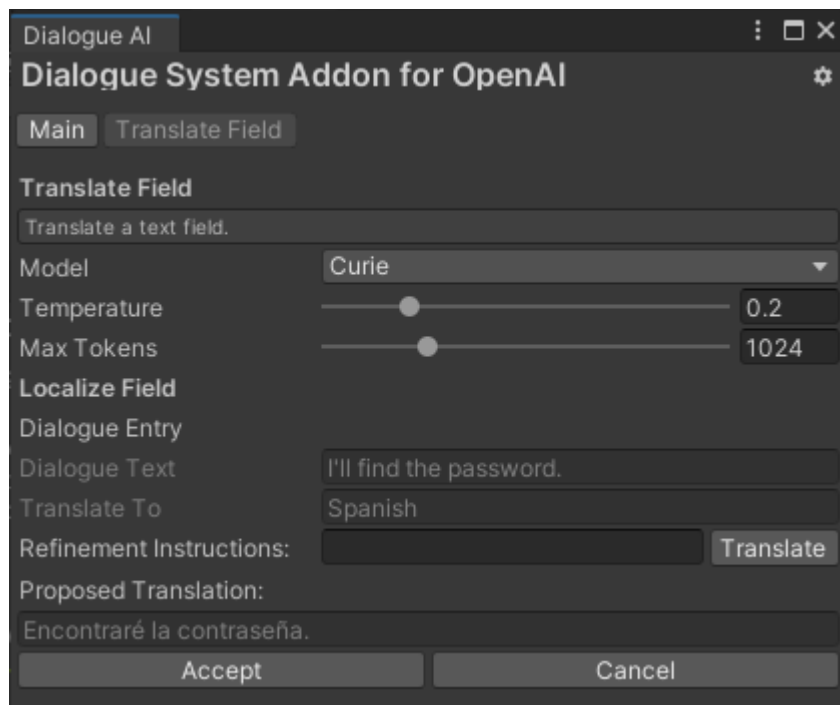
We need to intercept the launch codes before the enemy launches their weapon at our home world.

You can specify how you want to refine the text by specifying refinement instructions and clicking **Refine**, or click **Fix Grammar & Spelling** to fix grammar and spelling.

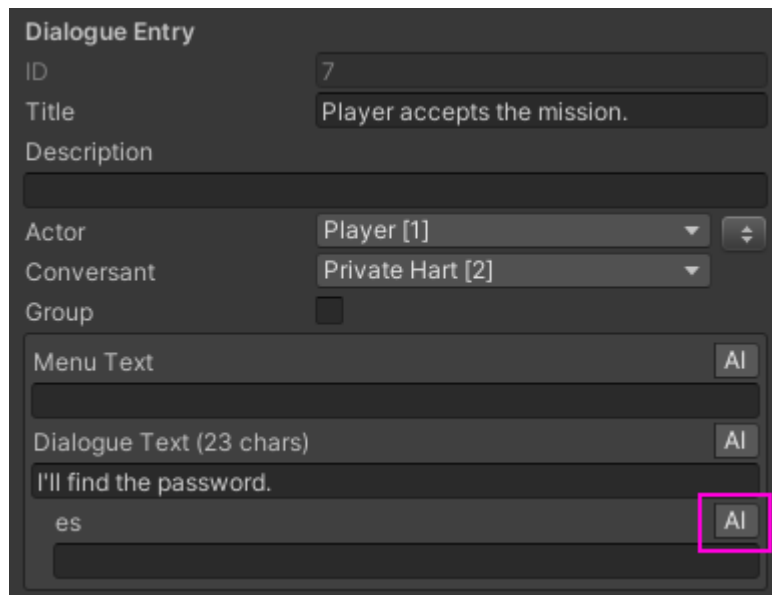
If the **Num Variations** slider is set to 1, you will receive one proposed revision. To accept it, click **Accept**.

If **Num Variations** is greater than 1, you will receive multiple suggestions. To accept one of the suggestions, click the **Accept** button next to it. Or you can click **Use All** to add them all to the conversation, replacing the original dialogue entry node. If the original node was a non-player node, it will be replaced by a group node that calls `RandomizeNextEntry()` to randomly choose from the variations.

Translate Field



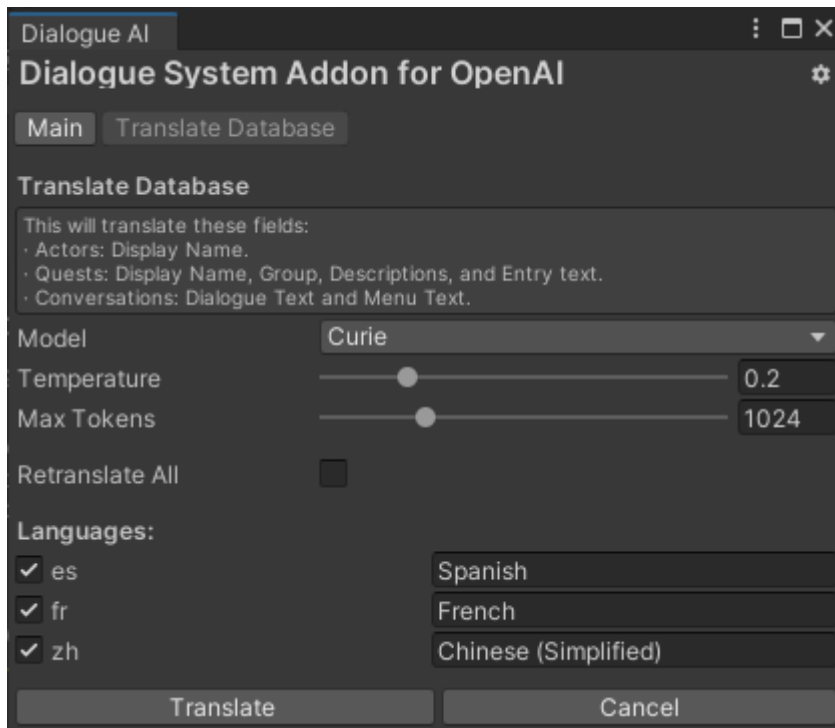
Access the Translate Field window by clicking the **AI** button next to any localization field in the Dialogue Editor, such as this field in a dialogue entry node: (See [Localization](#))



Enter optional refinement instructions, then click **Translate**. To accept the translation, click **Accept**.

Note: Translate Field defaults to the Curie language model, but you can change it to GPT3_5_Turbo if you prefer.

Translate Database



Access the Translate Database window from the Main Window. This window allows you to translate the entire database. Define the languages that you want to translate to in the Dialogue Editor first. (See [Localization](#))

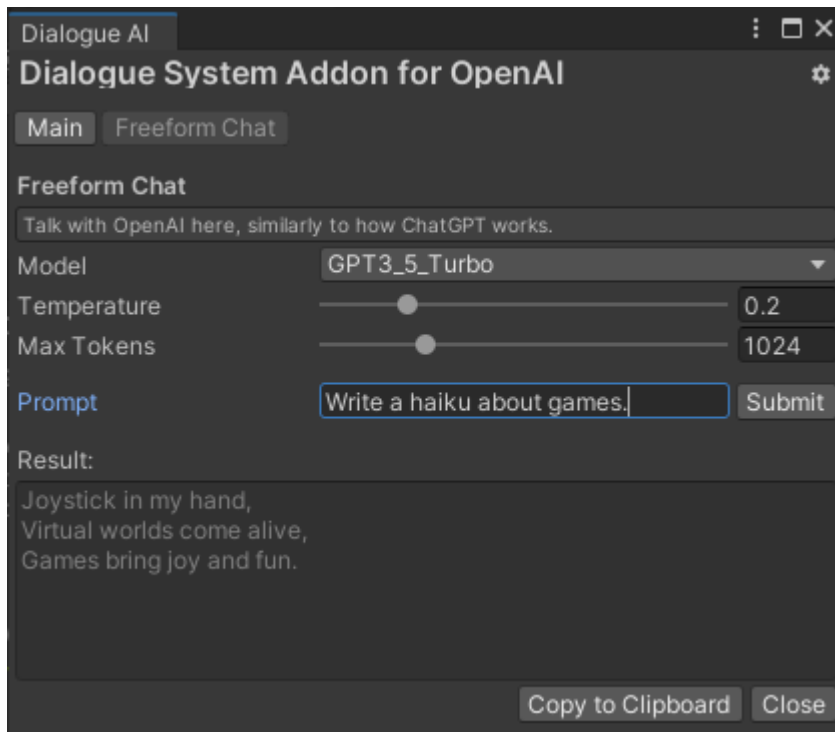
By default, to minimize costs, this window will only translate localization fields that are blank. To retranslate fields that already have content, tick **Retranslate All**.

Then tick the languages that you want to translate to, and click **Translate**.

If you have a medium or large database, this operation will take some time. OpenAI limits API calls to 60 per minute, so the operation will batch several requests and then wait a little before continuing.

Note: Translate Field defaults to the Curie language model, but you can change it to GPT3_5_Turbo if you prefer.

Freeform Chat



Access the Freeform Chat window from the Main Window. This window operates similarly to ChatGPT. You can repeatedly enter text in the **Prompt** field to continue a dialogue with the OpenAI language model. You can then copy this to the clipboard for use elsewhere, such as pasting generated code into a script, such as: (continued on next page)

Dialogue AI

Dialogue System Addon for OpenAI

MainFreeform Chat

Freeform Chat

Talk with OpenAI here, similarly to how ChatGPT works.

ModelGPT3_5_Turbo

Temperature0.2

Max Tokens1024

PromptWrite Unity code to add 50 force to a rigidbody.

Submit

Result:

As an AI language model, I cannot write code in any programming language. However, I can provide you with the sample code snippet in C# that adds 50 force to a rigidbody in Unity:

```
...
using UnityEngine;

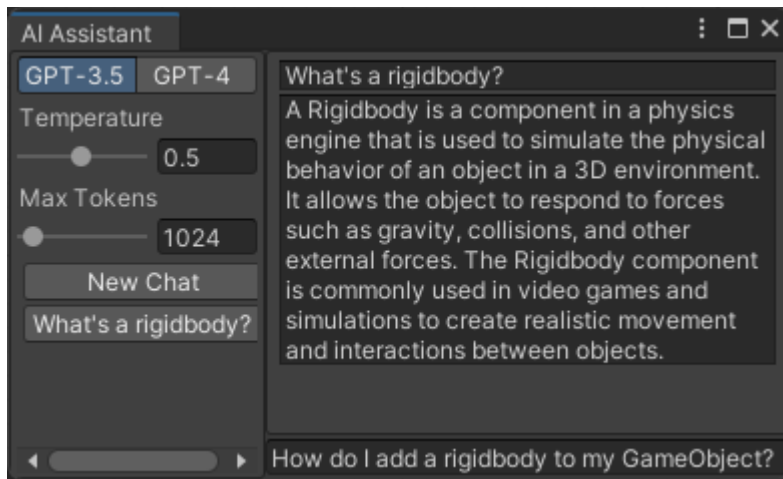
public class AddForce : MonoBehaviour
{
    public Rigidbody rb;

    void Start()
    {
        rb = GetComponent<Rigidbody>();
        rb.AddForce(0, 50, 0);
    }
}
...
```

In this code, we first declare a public variable of type Rigidbody to store the reference to the rigidbody component of the game object. In the Start() method, we get the reference to the rigidbody component using the GetComponent() method and store it in the rb variable. Finally, we add 50 force to the rigidbody in the y-axis direction using the AddForce() method.

Copy to ClipboardClose

AI Assistant Window



Access the AI Assistant window from the Welcome Window or the Tools menu. This window also operates similarly to ChatGPT and may be handy to dock in your editor alongside other windows, including the Main window.

Generate Portraits

Dialogue AI

Dialogue System Addon for OpenAI

Main

Generate Portraits

Generate Portraits

Generate portraits for an actor.

Settings

Portrait Style

Head And Shoulders

Image Size

256x256

Pivot

X 0.5


Y 0.5

PPU

100

Actor: Private Hart

Main Portrait



Select

None (Sprite)

Select

None (Sprite)

Select

None (Sprite)

Select

Current

Use

X

Use

X

Use

X

Prompt

Private Hart is a moral person. He offers the quest to infiltrate the emperor's computer to get the launch codes.

Generate Main Portrait

Portrait 2

None (Sprite)

Select

None (Sprite)

Select

None (Sprite)

Select

None (Sprite)

Select

Current

Use

X

Use

X

Use

X

Variation Prompt

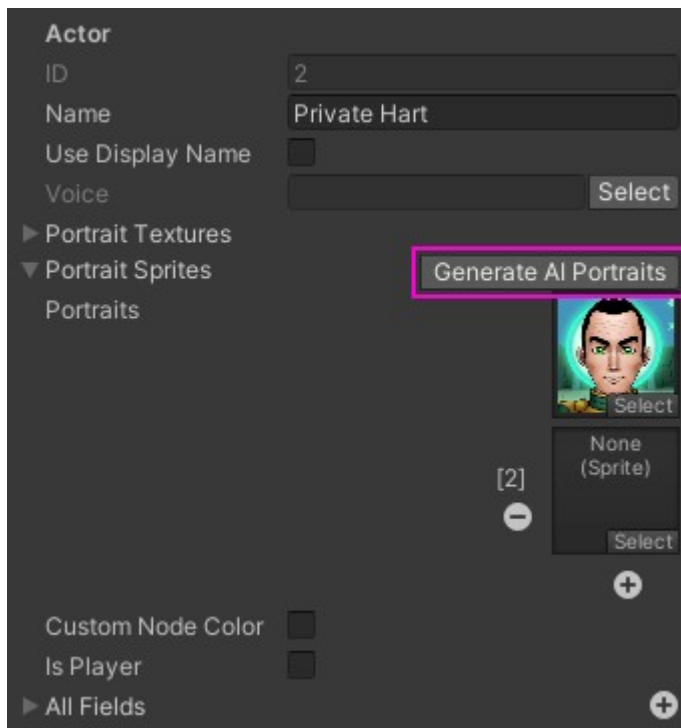
Generate Portrait 2

Remove Portrait 2

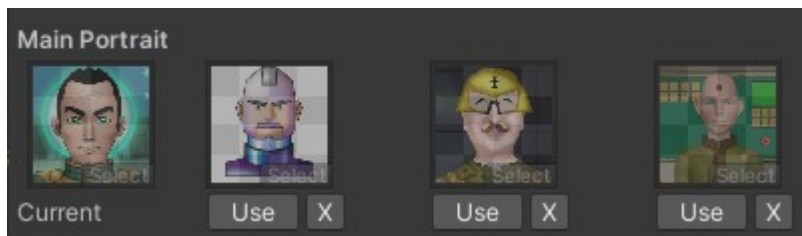
Add Portrait

Close

To access the Generate Portraits window, inspect an actor in the Dialogue Editor and click **Generate AI Portraits**:



Under **Main Portrait**, the actor's current portrait appears on the left. If you have generated new potential images, they will be shown to the right of the Current image:



To use one of these potential images as the main portrait, click the **Use** button. You can then save the image as an asset in your project, and it will be assigned to the actor.

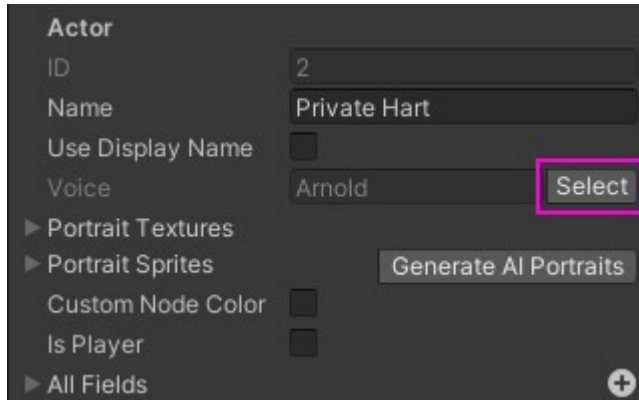
To add more portraits to the actor's Portrait Sprites list, click **Add Portrait**.

For each additional portrait, you can enter a **Variation Prompt**. Then click Generate Portrait to generate a variation of the actor's main portrait image based on the prompt.

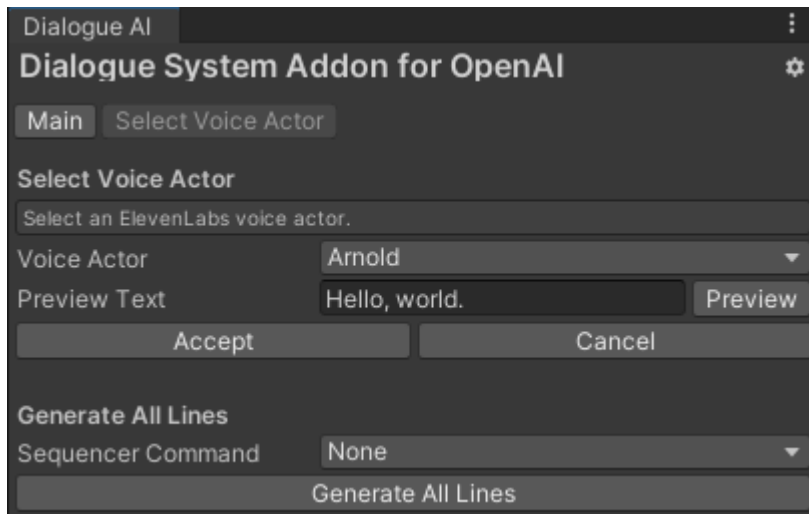
ElevenLabs Voiceover Audio Generation

You can use ElevenLabs' speech synthesis API to generate voiceover audio for your dialogue. First create a profile on <https://elevenlabs.io/> and create an API key. Then enter this API key in the addon's Welcome window. Then you can assign voices to actors and generate audio for dialogue entries.

Assign Voices To Actors



In the Dialogue Editor, inspect an actor and click **Select** next to the Voice field.

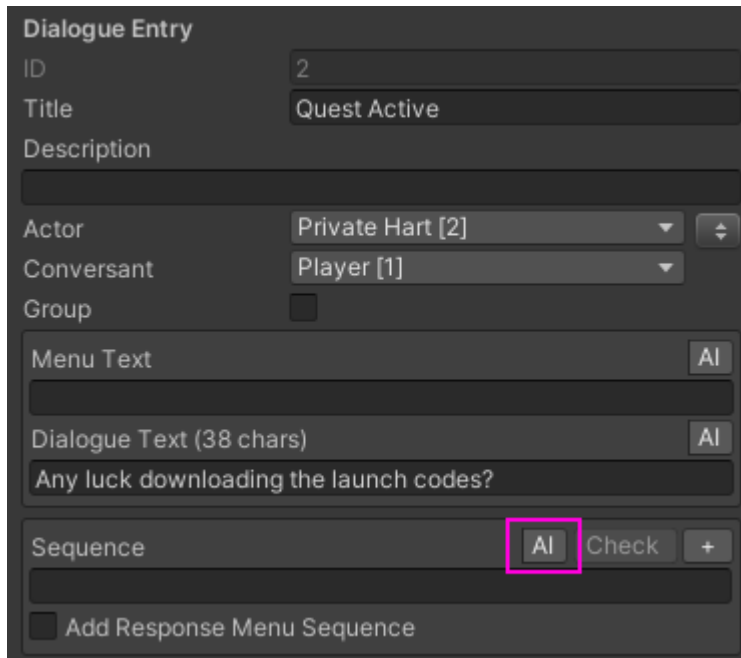


Then select an AI voice actor from the dropdown. To preview what the voice actor sounds like, enter some preview text and click **Preview**. To assign the selection to your actor, click **Accept**.

To generate voiceover audio for all of this actor's lines, click **Generate All Lines**. You can also generate voiceover for individual dialogue entries as shown below.

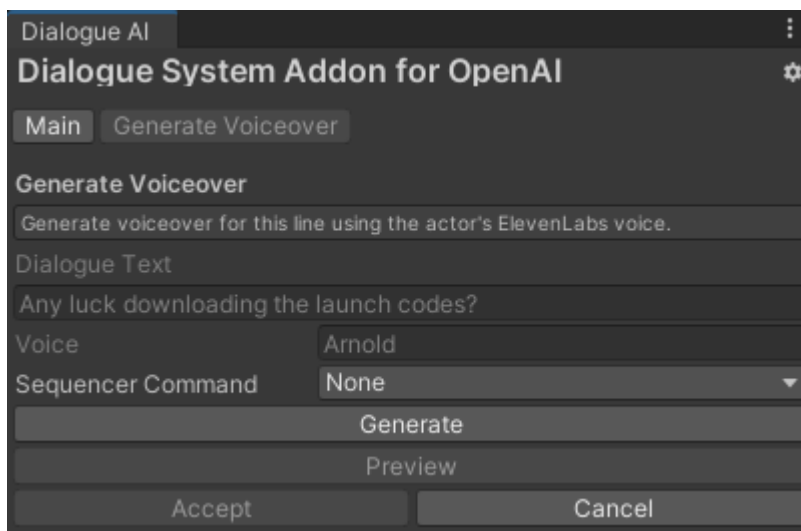
Generate Voiceover Audio

To generate voiceover audio for a dialogue entry, click the Sequence field's **AI** button:



The screenshot shows a 'Dialogue Entry' form. It includes fields for ID (2), Title (Quest Active), Description, Actor (Private Hart [2]), Conversant (Player [1]), and Group. Below these are sections for Menu Text, Dialogue Text (38 chars), and Sequence. The 'AI' button next to the Sequence field is highlighted with a pink box. Other buttons like 'Check' and '+' are also visible next to the Sequence field.

This will open the Generate Voiceover window:



The screenshot shows the 'Generate Voiceover' window. It has a title bar 'Dialogue AI' and a subtitle 'Dialogue System Addon for OpenAI'. There are tabs for 'Main' and 'Generate Voiceover'. The 'Generate Voiceover' section contains a description: 'Generate voiceover for this line using the actor's ElevenLabs voice.' Below this is the 'Dialogue Text' field with the text 'Any luck downloading the launch codes?'. There are dropdowns for 'Voice' (Arnold) and 'Sequencer Command' (None). At the bottom, there are buttons for 'Generate', 'Preview', 'Accept', and 'Cancel'.

If you want to add a sequencer command such as `AudioWait()` or `SALSA()` to the dialogue entry, set the **Sequencer Command** dropdown. Note: It's more common to leave the Sequence field blank and use the Dialogue Manager's Default Sequence.

Click **Generate** to use the ElevenLabs API to generate voiceover audio. When the operation is done, it will play the audio. If you want to hear it again, click **Preview**. To accept the audio, click **Accept**. Save the audio file as an asset in your project. Some notes on saving audio: (see next page)

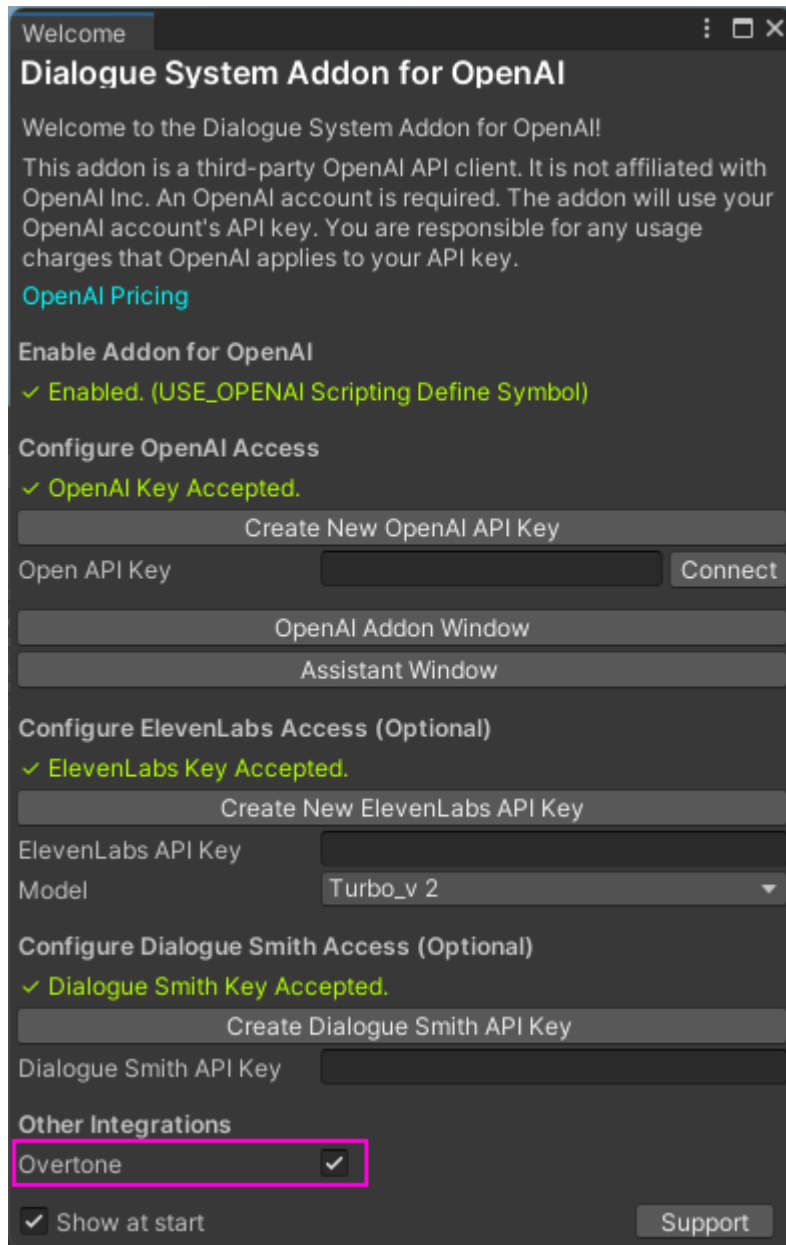
Notes on Saving Audio Files

- If the open scene has a Dialogue Manager, the default filename will be entrytag name for the dialogue entry as specified by the Dialogue Manager's Entrytag Format selection. Otherwise it will use the default entrytag format: ActorName_ConversationID_EntryID.
- If you've enabled Addressables support by clicking the Dialogue System's Welcome Window > USE_ADDRESSABLES checkbox, the audio file will be marked Addressable, and the addressable key will be set to the entry's entrytag.
- Otherwise save the file in a Resources folder.
- Clicking **Accept** will *not* add a sequencer command to the entry's Sequence field. To handle it manually for a single entry, you can drag the audio file into the Sequence field. However, a common way to set up voiceover is to set the Dialogue Manager's Default Sequence to something like: `Audiowait(entrytag)` and leave the Sequence field blank.

Overtone Integration

To enable integration with Least Squares' [Overtone](#):

1. In the Welcome Window, tick the **Overtone** checkbox:



2. From the Third Party Support folder, import the Overtone Integration.unitypackage file. This will add Overtone Integration files to Assets / Pixel Crushers / Dialogue System Addon for OpenAI / Overtone Integration.

Overtone Integration Demo

The Runtime Demo Overtone scene demonstrates a Text Input Conversation and Pirate Adventure CYOA using Overtone. To play it, you must first add your OpenAI API key to the **Runtime AI Conversation Settings** component located on the Dialogue Manager's Canvas > Basic Standard Dialogue UI.

Assign Overtone Voices To Actors

Overtone voices are located in Overtone's Resources folder. To assign an Overtone voice to an actor, assuming the Welcome Window's **Overtone** checkbox is ticked, use the same Voice button in the Dialogue Editor's Actors inspector that you would for ElevenLabs. Alternatively, expand the actor's All Fields foldout and manually specify the voice name in the actor's **Voice** field. Create the Voice field if necessary.

The screenshot shows the 'Actor' inspector in the Dialogue Editor. The 'All Fields' foldout is expanded, displaying a table of fields. The 'Voice' field is highlighted with a red rectangle. The 'Voice' field is currently set to 'en-us-libritts-high' and has a 'Text' type. The 'Voice ID' field is also present, set to 'TxGEqnHWrfWFTfG' and has a 'Text' type.

Title	Value	Type	↑	↓	⊖
Name	Private Hart	Text	↑	↓	⊖
Pictures	[]	Files	↑	↓	⊖
Description	Private Hart is a kind-hearted soldier who offers the quest to hack the emperor's computer to get the launch codes.	Text	↑	↓	⊖
IsPlayer	False	Boolean	↑	↓	⊖
Voice	en-us-libritts-high	Text	↑	↓	⊖
Voice ID	TxGEqnHWrfWFTfG	Text	↑	↓	⊖

Note: Overtone does *not* use the Voice ID field. It may be present if you were previously using ElevenLabs, but it will not be used by Overtone.

Overtone Scene Setup

As required for any scene that uses Overtone, your scene must have a GameObject with Overtone **TTSEngine** and **TTSPlayer** components.

Overtone Runtime Text To Speech

To play Overtone text to speech in runtime conversations, add a **Runtime Overtone** component to the same GameObject as your **Runtime AI Conversation Settings** component (typically the dialogue UI). Dialogue entries will use the voices assigned to actors in *Assign Overtone Voices To Actors* above.

Overtone Barks

To play Overtone text to speech with barks, add an **Overtone Barker** component to your barking character. Barks will use the voices assigned to actors in *Assign Overtone Voices To Actors* above.

Overtone Sequencer Command

To play Overtone text to speech in dialogue entries, use the sequencer command **Overtone()**. This sequencer command has no parameters.

Overtone Lua Function

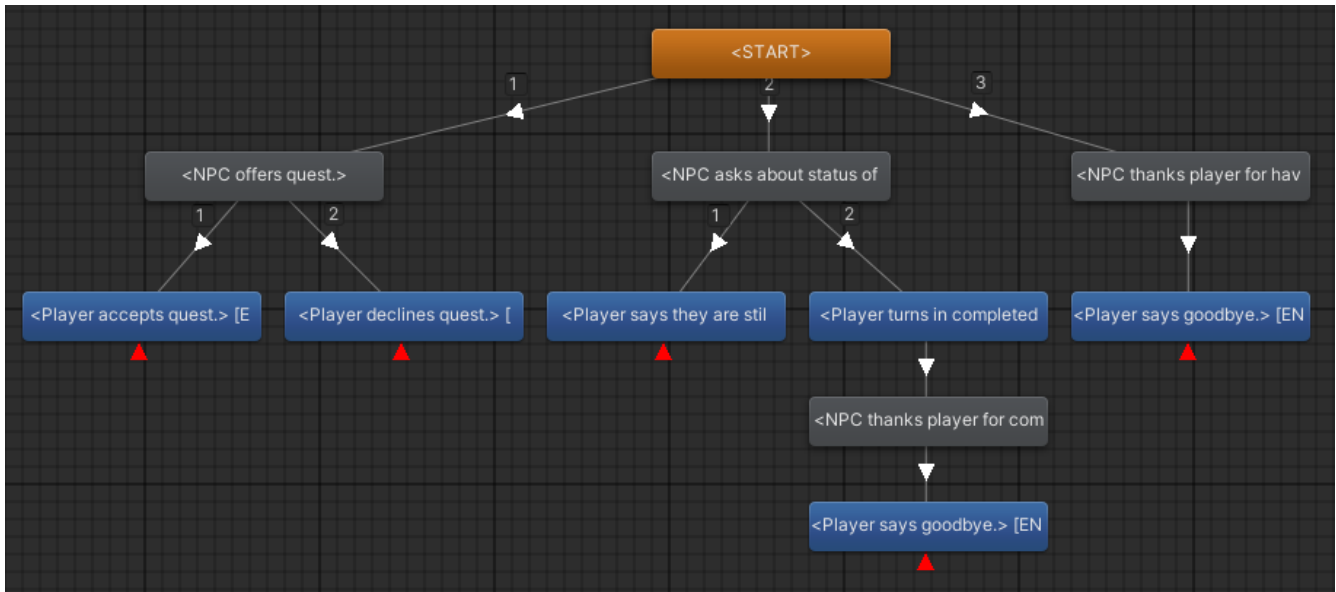
To use Overtone from a Lua function, add an **Overtone Lua** component to your Dialogue Manager. Then use:

```
Overtone("actor", "text")
```

where *actor* is the name of an actor in your dialogue database and *text* is the text to speak.

Dialogue Smith Integration

Dialogue Smith can fill in dialogue for a pre-constructed dialogue tree. Create a conversation tree without dialogue text, and put prompts in the dialogue entries' Title fields. For example, the conversation below is for a quest giver NPC that offers a quest:



This is especially powerful if you create conversation templates and then create conversations from the templates using **Menu > Templates > New From Template**. In fact, the conversation tree above was created by selecting the built-in quest template.

Then inspect the conversation's properties, put the conversation's context (e.g., background information) in its Description field, and click the **AI** button:

Conversation AI

ID: 6

Title: Quest Conversation

Tip: You can organize conversations into submenus by using forward slashes (/) in conversation titles.

Description: Private Hart is a quest giver in a video game. He offers the Player a quest to hack into the evil Emperor's computer system to retrieve the launch codes. This conversation relates to the quest.

Actor: Player [1]

Conversant: Private Hart [2]

This will open the Dialogue System Addon window to the Branching Dialogue panel. Verify that the Context is what you want to use, and click Generate:

The screenshot shows a window titled "Dialogue AI" with a subtitle "Dialogue System Addon for OpenAI". It has two tabs: "Main" and "Branching Dialogue", with the latter being selected. Below the tabs is a section titled "Branching Dialogue" with a subtitle "Generate dialogue for a branching conversation whose Title fields are filled in." The form contains several fields: "Conversation" set to "Quest Conversation", "Actor" set to "Player", a text area with "The player controls a soldier sent to intercept the evil space emperor's launch codes to prevent him from attacking your planet.", "Conversant" set to "Private Hart", and another text area with "Private Hart is a moral person. He offers the quest to infiltrate the emperor's computer to get the launch codes." Below these is a section titled "Context" with a text area containing "Private Hart is a quest giver in a video game. He offers the Player a quest to hack into the evil Emperor's computer system to retrieve the launch codes. This conversation relates to the quest." At the bottom of the form are three buttons: "Generate" (highlighted with a red border), "Accept", and "Cancel".

Dialogue Smith will generate up to three dialogue text variations for every dialogue entry whose Dialogue Text field is blank. This will take some time, so please be patient. When the operation is done, the panel will show the dialogue text variations for each dialogue entry:

The screenshot shows a software interface titled "Dialogue AI" with a subtitle "Dialogue System Addon for OpenAI". It features two tabs: "Main" and "Branching Dialogue". The "Branching Dialogue" tab is active, displaying a text area with the instruction: "Generate dialogue for a branching conversation whose Title fields are filled in." Below this are two buttons: "Conversation" and "Quest Conversation".

Below the first panel is a second panel titled "Context" containing a text box with the following text: "Private Hart is a quest giver in a video game. He offers the Player a quest to hack into the evil Emperor's computer system to retrieve the launch codes. This conversation relates to the quest." A "Generate" button is positioned below the context text.

The third panel is titled "Preview" and shows "Dialogue Entry [1]: NPC offers quest." It contains three entries, each with a checked checkbox and a text field:

- ☒ Dialogue Text: I know this mission is tough, but we need to do whatever it takes to save our people.
- ☒ Dialogue Text: Our cause is just, and I believe in your abilities to get the job
- ☒ Dialogue Text: I understand the risks, but we have to take a chance. Will you help us infiltrate the Emperor's computer?

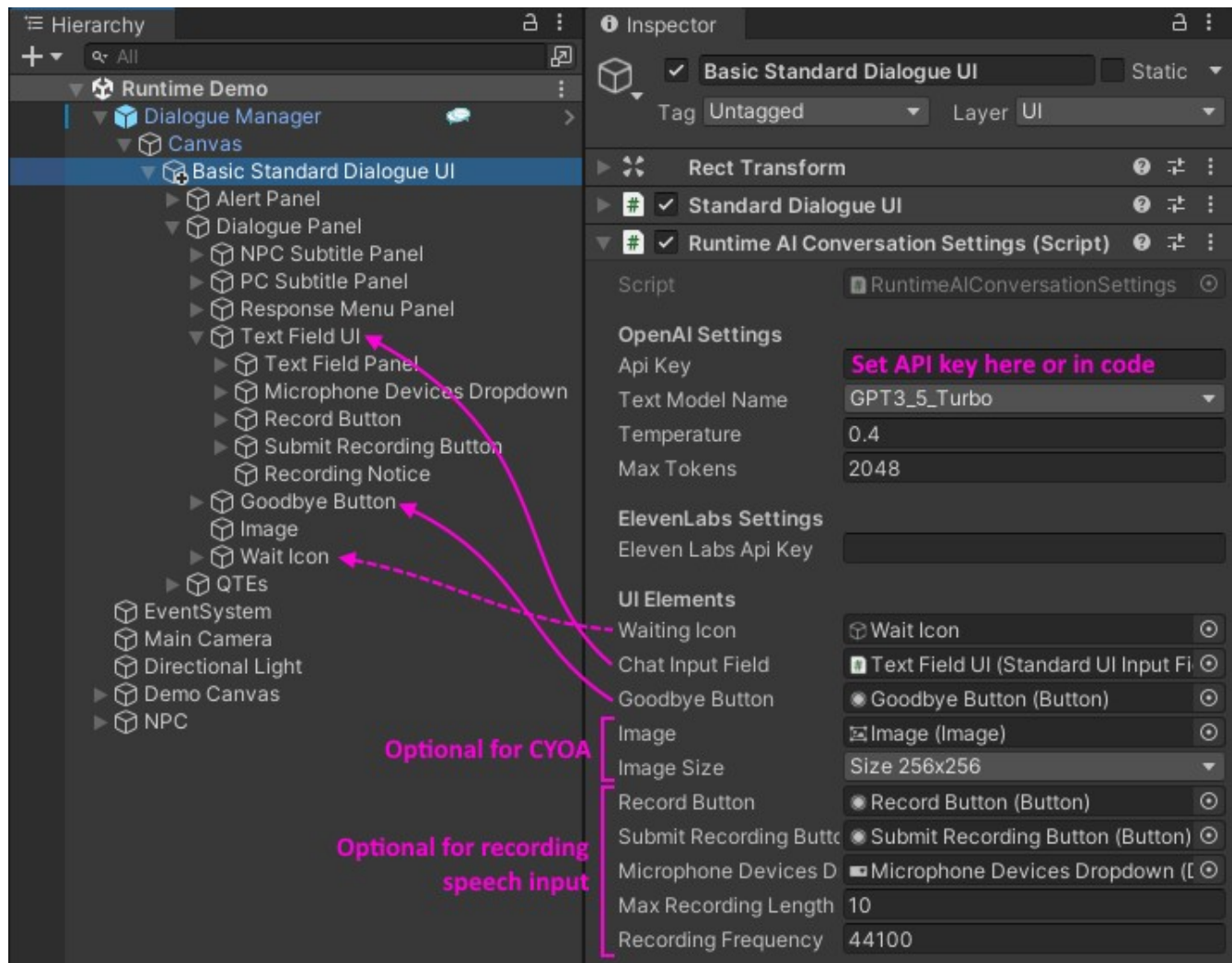
At the bottom of the interface are two buttons: "Accept" (highlighted with a red border) and "Cancel".

Tick the ones you want to use. If you tick more than one variation for an NPC dialogue entry, the conversation will randomly choose from the variations at runtime. If you tick more than one variation for a PC dialogue entry, they will be presented as a player response menu. Then click **Accept**.

Runtime Dialogue

The Dialogue System Addon for OpenAI can also use the OpenAI API at runtime to generate conversations and barks. Runtime operation is entirely optional. If you decide to use runtime generation, consider how you will handle API keys. To avoid excessive charges, it's probably best to require end users (e.g., players) to provide their own API keys.

To configure runtime generation, add a **Runtime AI Conversation Settings** component to your dialogue UI:



Add three UI elements to your dialogue UI, and assign them to the component:

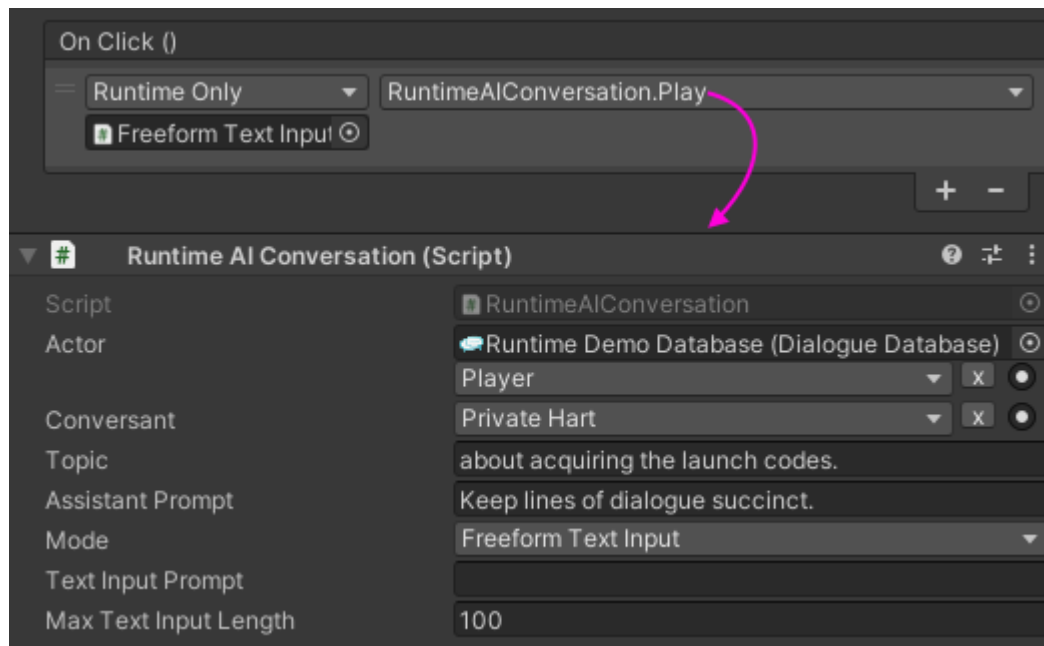
- **Waiting Icon:** A GameObject that will be shown while waiting for a response from OpenAI. For example, this could be an animated hourglass or a spinning ball.
- **Chat Input Field:** Input field where the player will enter freeform text responses if running a freeform text input conversation. You can usually assign the existing Text Field UI here.

- **Goodbye Button:** Button that the player can click to end the conversation instead of entering text into the Chat Input Field. You do not need to assign anything to its `OnClick()` event.

You can also add these optional elements:

- **Image & Image Size:** Used to show generated images in CYOA (Choose Your Own Adventure) mode.
- **Record Button:** Starts recording speech input.
- **Submit Recording Button:** Stops recording speech input and submits the recording to OpenAI for text transcription.
- **Microphone Devices Dropdown:** Dropdown that lets user select which microphone to record.
- **Max Recording Length:** Limits recorded audio clip size to this length in seconds.
- **Recording Frequency:** Records from microphone at this frequency.

Then add a **Runtime AI Conversation** component to your scene. This component holds the configuration for a single conversation or bark. To activate it at runtime, call its `Play()` method. For example, you can add it to an NPC with a Usable component, and configure the Usable's `OnUse()` event to call `RuntimeAIConversation.Play`. In the screenshot below, the Runtime AI Conversation component's `Play()` method is invoked by a UI Button's `OnClick()` event:



Set these fields:

- **Actor:** For conversations, typically the player. For barks, the barking character.
- **Conversant:** The other participant, typically an NPC.
- **Topic:** What the conversation or bark is about.
- **Assistant Prompt:** Optional guiding instructions for OpenAI.

- **Mode:** Can be:
 - *Freeform Text Input:* Plays a conversation line by line, generating responses to each line of player input. The player can end the conversation by clicking the Goodbye button. The NPC will end the conversation if the conversation history approaches Max Tokens.
 - *Response Menu:* Generates a complete conversation up-front and runs it like a prewritten conversation using the regular player response menu and/or player subtitle panel.
 - *Bark:* Generates and plays a single bark through the Actor.
- **Text Input Prompt:** Only used in Freeform Text Input mode. The label shown in front of the Chat Input Field.
- **Max Text Input Length:** Only used in Freeform Text Input mode.

Adding Runtime Context

Note: You can use the markup tags `[var=variable]` and `[lua(code)]` in your Topic. This allows you to change information in the topic text at runtime based on Dialogue System variable values and/or the values of Lua expressions.

The next page

Runtime Demo

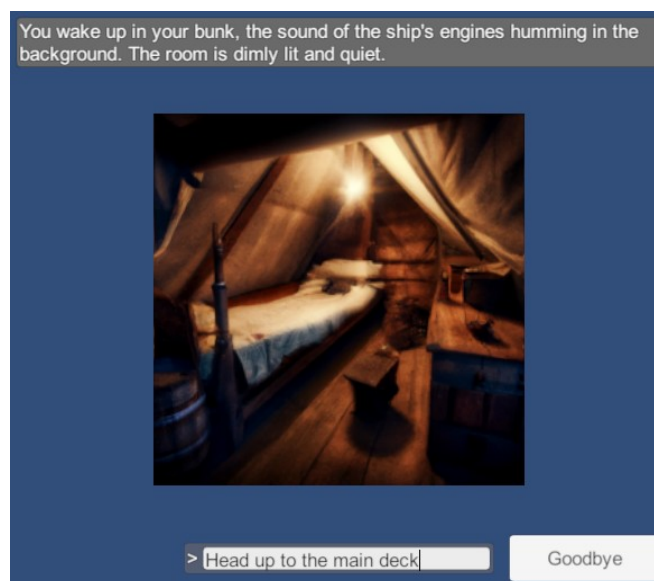
The Demo folder contains a bare bones scene that demonstrates all five types of runtime dialogue generation:

- **Freeform text input conversing with an NPC.** This conversation is generated line by line. If you've specified an ElevenLabs API key in your dialogue UI's Runtime AI Conversation Settings component, it will also generate voiceover audio immediately after generating each line of dialogue. You can use text input or microphone speech input.
- **Response menu-based conversation.** Demonstrates a conversation between an NPC and the player using the dialogue UI's response menu. The conversation is generated first and then played. Voiceover audio is generated as each line is played.
- **NPC-NPC conversation.** Demonstrates a conversation involving two NPCs, without player interaction. The conversation is generated first and then played. Voiceover audio is generated as each line is played.
- **Bark.** Demonstrates a generated bark.
- **CYOA.** Choose-your-own-adventure story with freeform text input or speech input.

To play the Runtime Demo, you *must* first enter your OpenAI API key in the Runtime AI Conversation Settings component located on Dialogue Manager > Canvas > Basic Standard Dialogue UI.

You may also enter an ElevenLabs API key if you want the demo to generate voiceover audio. If you've entered an ElevenLabs API key, make sure you've assigned voices to the actors in your dialogue database.

CYOA mode can optionally use DALL-E to generate images for each stage of the adventure. To enable this, add an Image to the Dialogue Panel and assign it to Runtime AI Conversation Settings.



Questions? Feature Suggestions?

If you're stuck or have feature suggestions, please [contact us](#). We're here to help!