Unity Plugin

Integrate advanced conversational AI to create intelligent, interactive NPCs for your games.

Overview

Convai's Unity SDK provides you with all the tools you need to integrate conversational AI into your Unity projects. Convai offers specialized NLP-based services to build intelligent NPCs for your games and virtual worlds. Our platform is designed to seamlessly integrate with your game development workflow, enhancing the interactivity and depth of your virtual environments.

Key Features

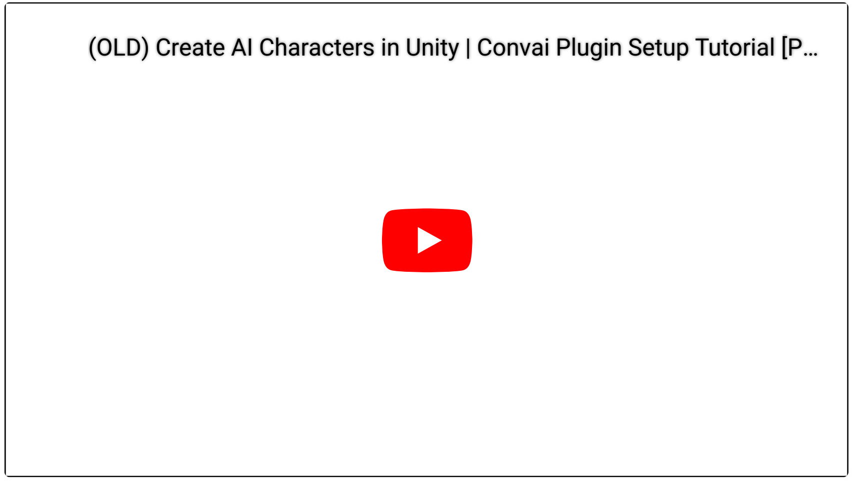
- **Conversational AI**: Leverage advanced NLP capabilities to create NPCs that can understand and respond to player input in natural, engaging ways.
- **Intelligent NPCs**: Build characters with dynamic dialogue and behaviors that adapt to player actions and the game world.
- **Easy Integration**: Our SDK is designed for quick and simple integration with your Unity projects, allowing you to focus on creating compelling gameplay experiences.
- **Cross-Engine Support**: In addition to Unity, Convai supports other popular game engines, ensuring broad compatibility and flexibility for your development needs.

Download the Convai Unity SDK from Unity Asset Store

This is the Core version of the plugin. It has a sample scene for anyone to get started. This version of the plugin only contains the basic Convai scripts and Character Downloader.

Visit convai.com for more information and support.

Quick Setup Tutorial


(OLD) Create AI Characters in Unity | Convai Plugin Setup Tutorial [P...

Unity Plugin Quick Setup Video

Pre-Requisites

Review the prerequisites for integrating Convai with Unity. Ensure seamless setup and functionality.

Unity Version

The Convai Unity SDK supports a minimum of Unity 2022.3.x or later.

You should have Git installed locally on your system.

Skills and Knowledge

Before integrating the Convai SDK, you should be comfortable with the following:

1. **Importing Packages**: Know how to import external packages into a Unity project.
2. **Unity Editor**: Be proficient in navigating the Unity Editor interface.
3. **Animations**: Understand how to add and handle animations for assets.
4. **Programming in C#**: Have a basic experience programming Unity scripts in C#.
5. **Script Integration**: Be capable of adding scripts to a game object.
6. **Building and Deployment**: Know how to build and deploy an application to your chosen platform.

Having these skills will ensure a smooth integration and optimal use of the Convai Unity SDK in your projects.

Compatibility

Check Convai plugin compatibility with Unity. Ensure smooth integration with your development tools.

Unity Version

The minimum supported Unity version is **2022.3.x**. Earlier versions may not be compatible.

Supported Platforms

Tested Platform	Scripting Backend	API Level
Windows	MONO	.NET Standard 2.1 or .NET 4.x+
MacOS	MONO	.NET Standard 2.1 or .NET 4.x+
Android	IL2CPP	.NET 4.x+
iOS	IL2CPP	.NET 4.x+
WebGL	MONO	.NET Standard 2.1 or .NET 4.x+
Oculus	IL2CPP	.NET 4.x+

Unity Version	API Level
2022.3 or Higher	.NET Standard 2.1 or .NET Framework

Disabling Assembly Validation

Downloads

Download Convai tools for Unity. Access the latest plugins and updates for AI integration.

Version	Features	Download Link
Unity Verified Solution	This is the Long-Term Support version of our core version. It contains all the necessary tools for adding conversational AI to your characters.	**Download here.**
WebGL	This plugin version should be used if you need to build for WebGL. Please ensure that Git is installed on your computer prior to proceeding.	**Download here.**

There are some limitations for WebGL version of the plugin, to learn about it, please go to Limitations of WebGL Plugin

Limitations of WebGL Plugin

Understand the limitations of the WebGL plugin for Unity with Convai. Optimize your development.

Size Constraints

iOS browsers impose strict limitations on the size of WebGL builds. These constraints are primarily due to:

- Memory Limits: iOS devices have limited available memory for web applications, which can affect the performance and feasibility of running large WebGL builds.
- Browser Storage Quotas: Safari and other iOS browsers restrict the amount of data that can be stored locally. This includes caching and Indexed DB, which are often used to store assets for WebGL builds.

Key Limitations

- Maximum Downloadable Asset Size: iOS browsers may restrict the size of individual downloadable assets. Large assets might fail to load, causing the application to break.
- Total Build Size: The total size of all assets combined should ideally be kept under 50-100 MB for smooth performance. Exceeding this limit can lead to crashes or extremely slow loading times.
- Memory Usage: iOS devices typically have less RAM available compared to desktop environments. High memory usage by WebGL builds can result in frequent browser crashes.

Browser Compatibility

- Safari: The default browser on iOS, Safari, is generally the best option for WebGL builds, but it still has significant limitations compared to other desktop browsers.

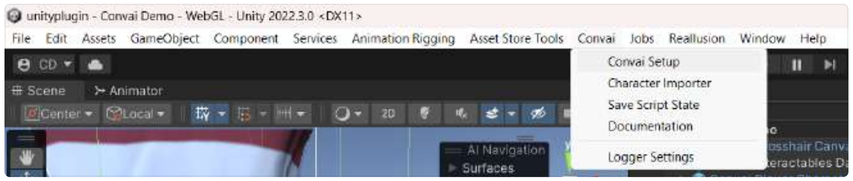
Setting Up Unity Plugin

Follow these instructions to setup the Unity Plugin into your project.

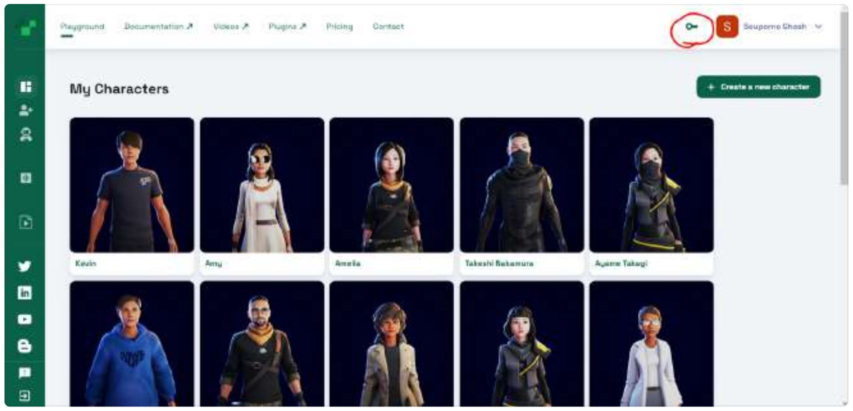
The file structure belongs to the Core version of the plugin downloaded from the documentation.

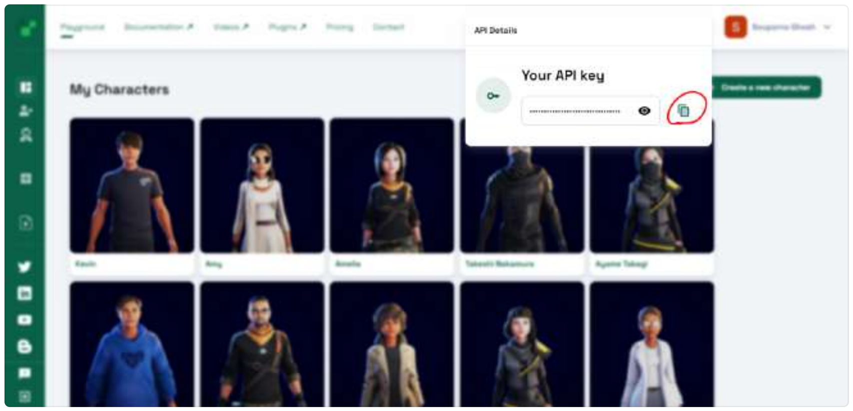
Setting up Unity Plugin

In the Menu Bar, go the Convai > API Key Setup.

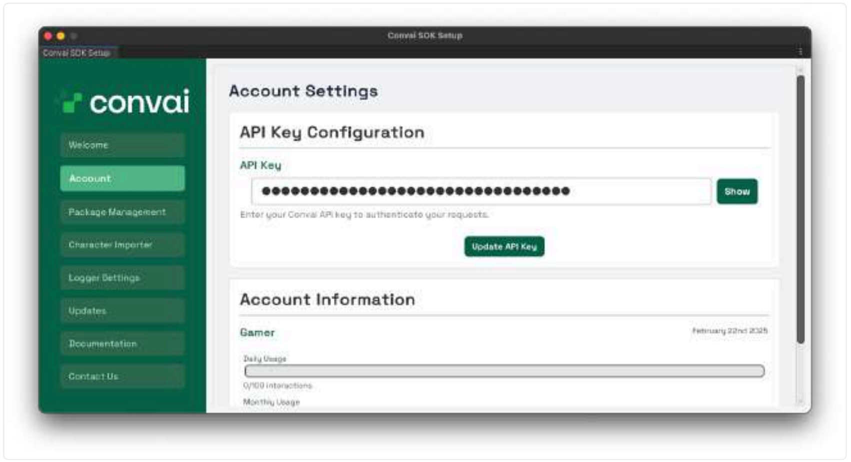

unityplugin - Convai Demo - WebGL - Unity 2022.3.0 <DX11>
File Edit Assets GameObject Component Services Animation Rigging Asset Store Tools Convai Jobs Reallusion Window Help
Convai Setup
Character Importer
Save Script State
Documentation
Logger Settings

Go to convai.com, and sign in to your Convai account. Signing in will redirect you to the Dashboard. From the dashboard, grab your API key.



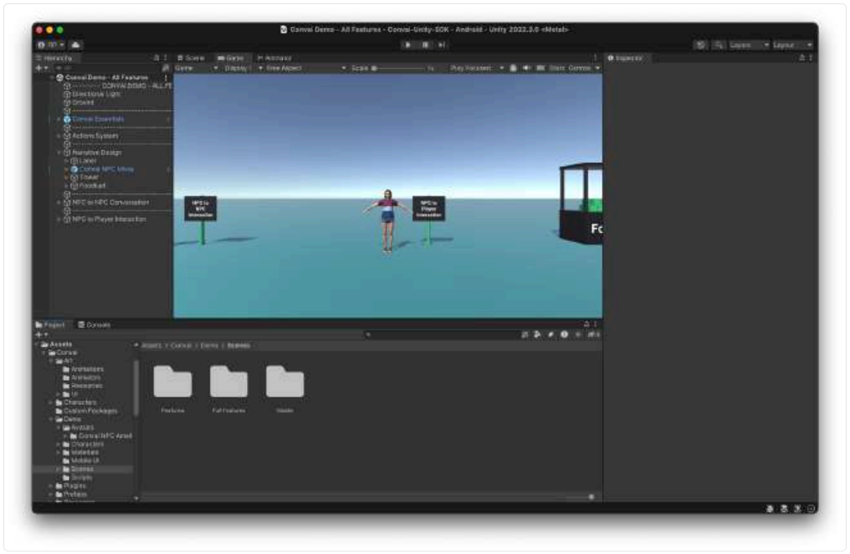


Enter the API Key and click begin.

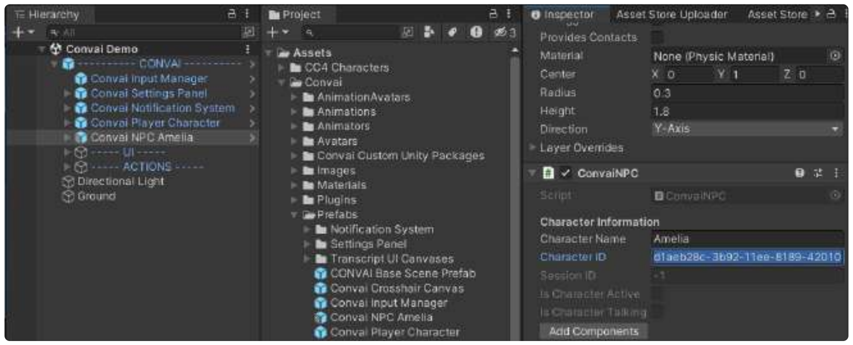


This will create an APIKey asset in the resources folder. This contains your API Key.

Open the demo scene by going to Convai > Demo > Scenes > Full Features



Click the Convai NPC Amelia and add the Character ID (or you can keep the default character ID). You can get the character ID for your custom character from this page Create Character. Now you can converse with the character. The script is set up so that you have to go near the character for them to hear you.

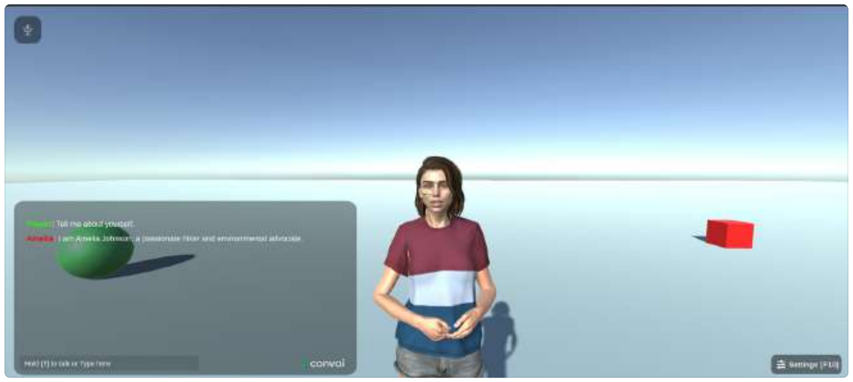


Provides Contacts
Material: None (Physic Material)
Center: X 0 Y 1 Z 0
Radius: 0.3
Height: 1.8
Direction: Y-Axis
Layer Overrides

ConvaiNPC
Script: ConvaiNPC

Character Information
Character Name: Amelia
Character ID: d1aeb28c-3b92-11ee-8189-42010
Session ID: -1
Is Character Active
Is Character Talking

Now you can test out the Convai Demo Scene and talk to the character present there. Her name is Amelia and she loves hiking!


Tell me about yourself.
I am Amelia Johnson, a passionate hiker and environmental advocate.
Hold [T] to talk or Type here
convai
Settings [F10]

You can open the Convai NPC Script to replicate or build on the script to create new NPCs.

Try to extend the ConvaiNPC.cs script instead of directly modifying it to maintain compatibility with other scripts

Troubleshooting Guide

Troubleshoot common issues with Convai's Unity plugin. Get solutions for seamless AI integration.

Common Issues (FAQ)

Q. I cannot see the Convai menu.

A. Please check if there are any errors in the console. Unity needs to be able to compile all the scripts to be able to display any custom editor menu options. Resolving all the console errors will fix this issue.

Q. There are a lot of errors on my console.

A. Primarily, two issues cause errors in the console that can stem from the Convai Unity Plugin. You can use the links below to fix them quickly.

1. Disable Assembly Validation
2. Missing Newtonsoft Json

Q. I am talking to the character, but I cannot see the user transcript and the character does not seem to be coherently responding to what I am saying.

A. This may indicate issues with the microphone. Please ensure that the microphone is connected correctly. You also need to ensure that the applications have permission to access the menu.

Microphone Permission Issues

Q. The animations for my characters are looking very weird.

A. The animation avatar that we are using might be incompatible with the character mesh. Fixing that can solve the issue.

Default Animations Incompatibility

Q. There are two Settings Panel Buttons in Mobile Transcript UI.

A. If you are using Unity 2021, unexpected prefab variant issues may arise. This is because Unity Mobile Transcript UIs are variants of the main transcript UI prefab. With changes in the Prefab system in Unity 2022, it works correctly in Unity 2022. If you are using Unity 2021, you may encounter issues with prefabs. You can remove the redundant Settings Panel Button to address this problem.

Q: The lipsync is very faint or not visible.

A: The animations that we are using may be modifying facial animations. Editing the animations to remove facial animations should fix any issues related to lipsync.

Animations have Facial Blendshapes

A: The script also needs the avatar to not be mapped to the jaw bone to be manipulate the jaw bones itself.

Jaw Bone in Avatar is not Free

Q: I'm facing security permission issues using the `grpc_csharp_ext.bundle` DLL inside the Unity Editor on MacOS

A: macOS's strict security measures can block certain external unsigned DLLs. To address this, you can manually allow the DLL in "Security & Privacy" settings, modify Gatekeeper's settings through Terminal, ensure correct file permissions for the DLL, check its settings in Unity, and update the Mac Configuration in Unity's Player Settings

macOS Permission Issues

Q: I'm not able to talk to my character after building my Unity project for macOS (Intel64+Apple Silicon builds), especially on Intel Macs

A: The issue is rooted in the `grpc_csharp_ext.bundle` used in Unity for networking. This DLL has separate versions optimized for Intel and Apple Silicon architectures. When trying to create a Universal build that serves both, compatibility problems arise, especially on Intel Macs. Presently, the best solution is to use Standalone build settings specific to each architecture.

Building for macOS Universal apps

Error Index

Follow this Table to navigate to our most common errors.

Name	Sample Error	Reason for Error	
Enabled Assembly Validation	`Assembly 'Assets/Convai/Plugins/Grpc.Core.Api/lib/net45/Grpc.Core.Api.dll' will not be loaded due to errors: Grpc.Core.Api references strong named System.Memory Assembly references: 4.0.1.1 Found in project: 4.0.1.2.`	Unity, by default, checks for exact version numbers for the included assemblies. For our plugin, this is not necessary, since we use the latest libraries.	Disable Assembly Validation
Missing NewtonSoft Json	`Assets\Convai\Plugins\GLTFUtility\Scripts\Spec\GLTFPrimitive.cs(8,4): error CS0246: The type or namespace name 'JsonPropertyAttribute' could not be found (are you missing a using directive or an assembly reference?)`	Our plugin needs Newtonsoft Json as a dependency. It is often present as part of Unity but occasionally, it can be missing.	Missing Newtonsoft Json

Name	Sample Error	Reason for Error	
Missing Animation Rigging	`Assets\Convai\Scripts\Utils\HeadMovement.cs (2,30): error CS0234: The type or namespace name 'Rigging' does not exist in the namespace 'UnityEngine.Animations' (are you missing an assembly reference?)`	We use the Animation Rigging package for Eye and Neck tracking. If Unity does not automatically add it, we need to add it manually from the package manager.	
Microphone Permission Issues	The microphone icon lights up but there is no user transcript in the chat UI. The character seemingly not replying to what the user is saying.	The plugin requires microphone access which is sometimes not enabled by default.	Microphone Permission Issues
Default Animations Incompatibility	The default animations that ship with the plugin seems broken. The hands seem to intersect with the body.	The animation avatar is incompatible with the character mesh.	Default Animations Incompatibility
Animations have Facial Blendshapes	The Lip-sync from characters are either not visible or are very faint.	Some types of animations control facial blendshapes. These animations prevent the lip-sync scripts to properly edit the facial blendshapes.	Animations have Facial Blendshapes
Jaw Bone in Avatar is not Free	The Lip-sync from characters are either not visible or are very faint.	The animation avatar for the character may be using the Jaw Bone. If we set the mapping free to none, the script will be able to manipulate the jaw bone freely.	Jaw Bone in Avatar is not Free
Mac Security Permission Issue	Security Permission Issues with `grpc_csharp_ext.bundle` DLL in Unity on MacOS.	MacOS's security protocols can prevent certain unsigned external DLLs, like `grpc_csharp_ext.bundle`, from functioning correctly in Unity.	macOS Permission Issues
Microphone Permission Issue with Universal Builds on Intel Macs in Unity	No Microphone access request pops up	Incompatibility between Intel and Apple Silicon versions of `grpc_csharp_ext.bundle` when attempting a	Building for macOS Universal apps

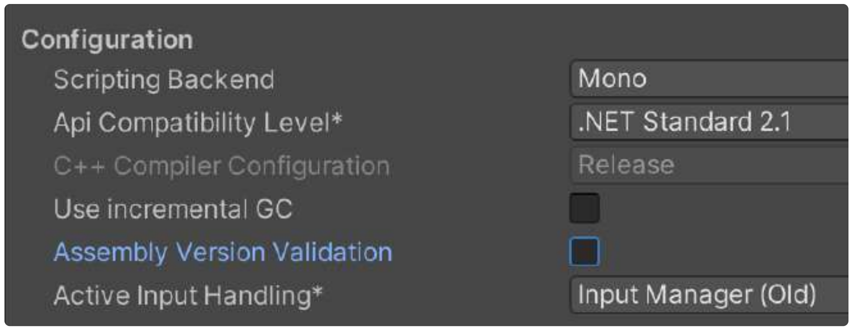
For any other issues, please feel free to reach out to support@convai.com or on our Discord Server.

Disable Assembly Validation

If you ever get an error that looks like this, disable the Assemble Version Validation in `Project Settings` > `Player` > `Other Settings`.

```
Assembly
'Assets/Convai/Plugins/Grpc.Core.Api/lib/net45/Grpc.Core.Api.dll' will
not be loaded due to errors:
Grpc.Core.Api references strong named System.Memory Assembly references:
4.0.1.1 Found in project: 4.0.1.2.
```

Ensure that Assembly Validation is disabled in `Project Settings` > `Player` > `Other Settings`.



Configuration
Scripting Backend: Mono
Api Compatibility Level*: .NET Standard 2.1
C++ Compiler Configuration: Release
Use incremental GC: ☐
Assembly Version Validation: ☐
Active Input Handling*: Input Manager (Old)

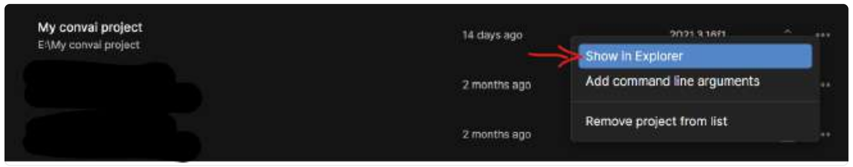
Restart the Unity project after unchecking the box should fix the issue.

Missing Newtonsoft Json

Fix missing Newtonsoft JSON issues in Unity with Convai. Resolve integration problems efficiently.

Our plugin has various scripts and dependencies that use Newtonsoft Json. If Newtonsoft Json is missing from the plugin, it could lead to a large number of errors as shown below:

Ensure that NewtonSoft.Json is present in your packages. Go to your project folder.



My convai project
E:\My convai project
14 days ago
[redacted]
2 months ago
2 months ago

Then navigate to Packages folder. In the Packages folder. Click on manifest.json. A json file containing the project dependacies should open up.

Add the Newtonsoft Json Package on top.

```
    "com.unity.nuget.newtonsoft-json": "3.2.1",
```

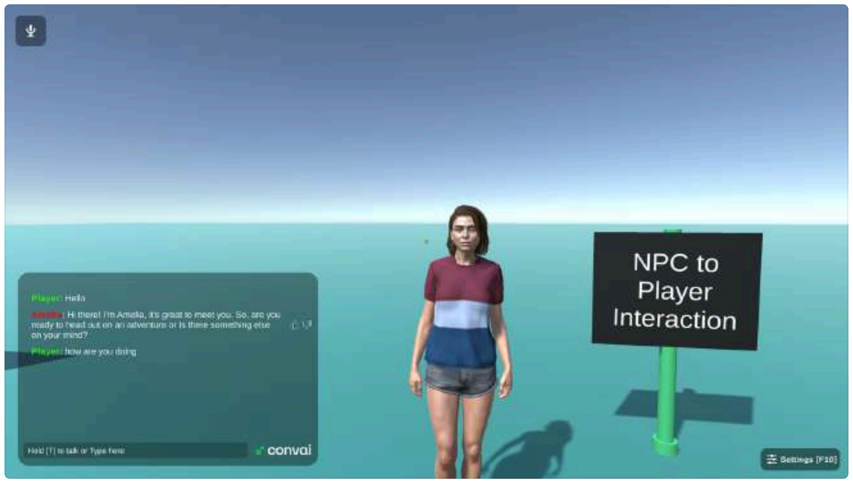
The final manifest.json should look like this.

```
{
    "dependencies": {
        "com.unity.nuget.newtonsoft-json": "3.2.1",
        "com.unity.animation.rigging": "1.1.1",
        "com.unity.ide.rider": "3.0.16",
        "com.unity.ide.visualstudio": "2.0.16",
        "com.unity.ide.vscode": "1.2.5",
        "com.unity.test-framework": "1.1.33",
        "com.unity.textmeshpro": "3.0.6",
        "com.unity.timeline": "1.6.4",
        .
        .
        .
    }
}
```

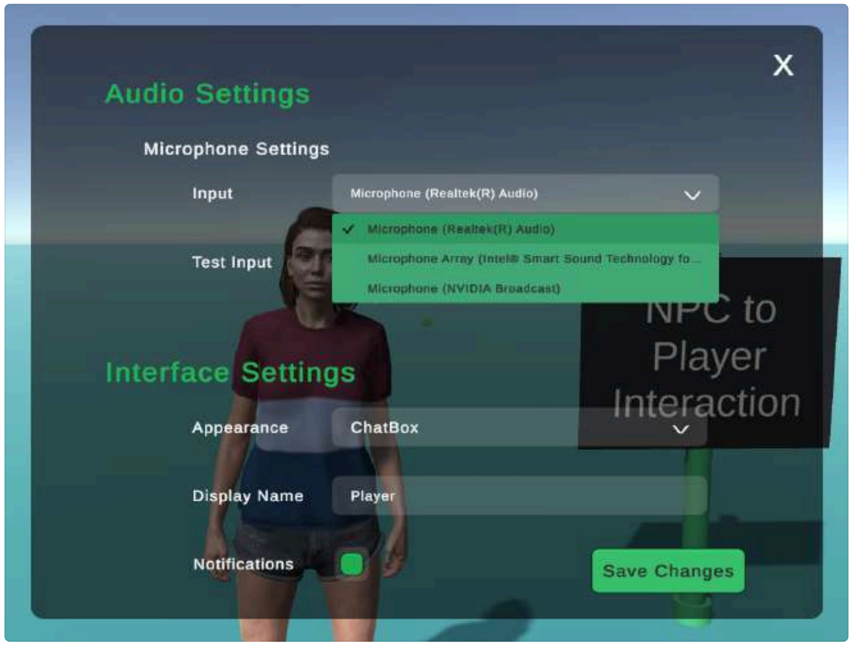
Microphone Permission Issues

Resolve microphone permission issues in Unity with Convai. Ensure smooth voice interactions.

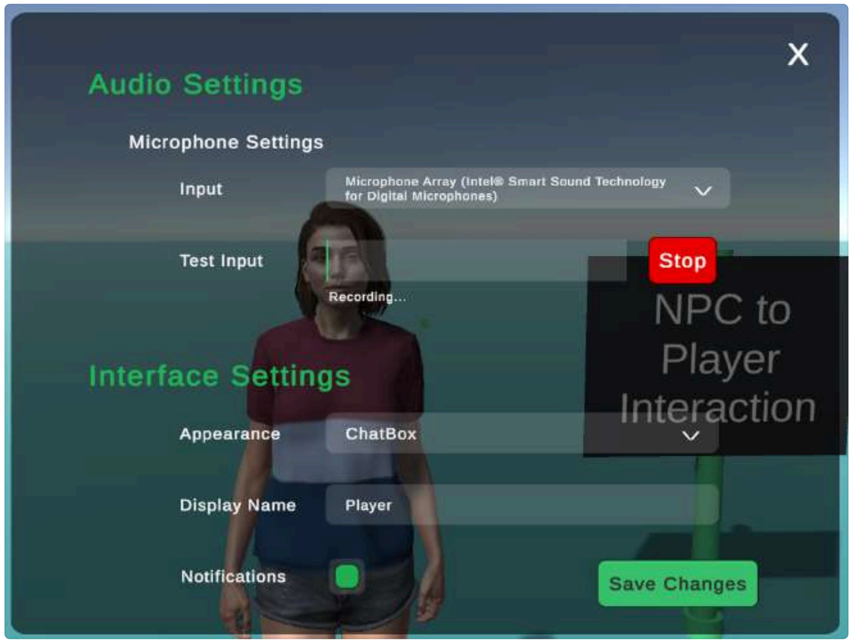
If you see the microphone indicator turning on in the top left corner but no user transcript in the chat UI and the character's response doesn't seem coherent to what you said, then it is likely that the game or Unity is not accessing the correct microphone or does not have sufficient microphone privilege. To fix this, please follow along.


Player: Hello
Amelia: Hi there! I'm Amelia, it's great to meet you. So, are you ready to head out on an adventure or is there something else on your mind?
Player: how are you doing

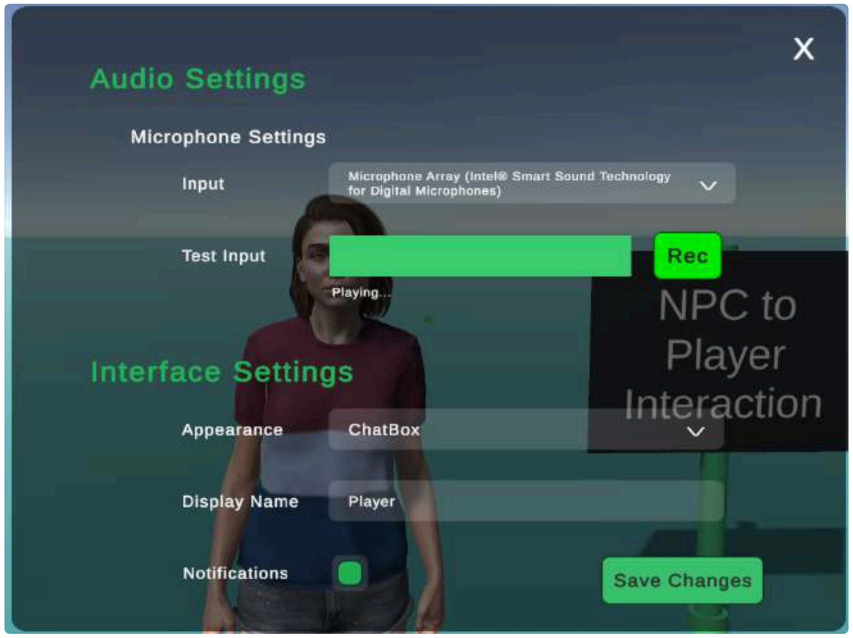
The microphone icon on the top left is on, indicating that mic is listening


Audio Settings
Microphone Settings
Input
Microphone (Realtek(R) Audio)
Microphone (Realtek(R) Audio)
Microphone Array (Intel® Smart Sound Technology fo...
Microphone (NVIDIA Broadcast)
Test Input
NPC to Player Interaction
Interface Settings
Appearance
ChatBox
Display Name
Player
Notifications
Save Changes
X

Select proper microphone device from the drop down list



Click on Record to make sure microphone is recording properly


Audio Settings

Microphone Settings

Input: Microphone Array (Intel® Smart Sound Technology for Digital Microphones)

Test Input: Playing...

Interface Settings

Appearance: ChatBox

Display Name: Player

Notifications: ✓

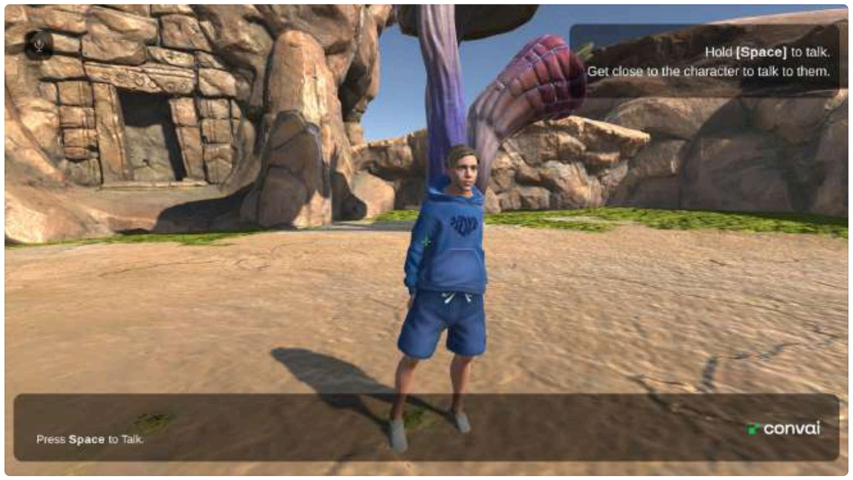
Press Stop to listen to the audio recorded through the selected mic

Check Notification System page to learn more about various issues related to microphone

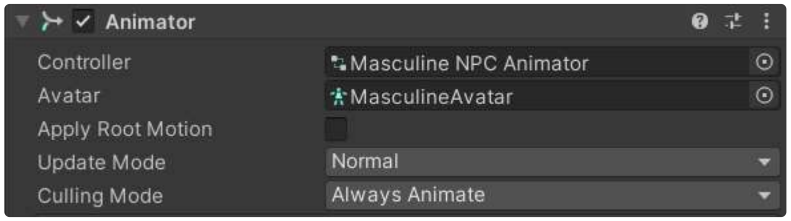
Default Animations Incompatibility

Fix default animation incompatibilities in Unity with Convai. Ensure smooth AI character animations.

If the default animations that ship with the animator look bugged such that the hand seems to intersect with the body, it could indicate an issue with the wrong animation avatar being selected.

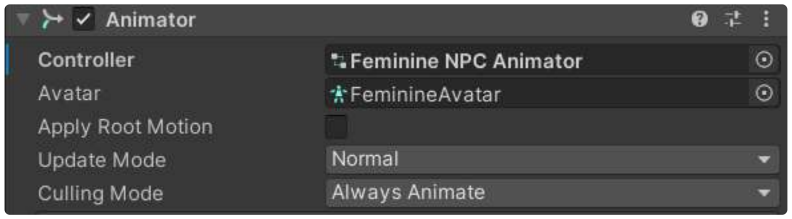


You can easily fix that by heading to the character's animator component and assigning the correct animator to the Avatar field.


Animator

Controller	Masculine NPC Animator
Avatar	MasculineAvatar
Apply Root Motion	
Update Mode	Normal
Culling Mode	Always Animate

For male avatars

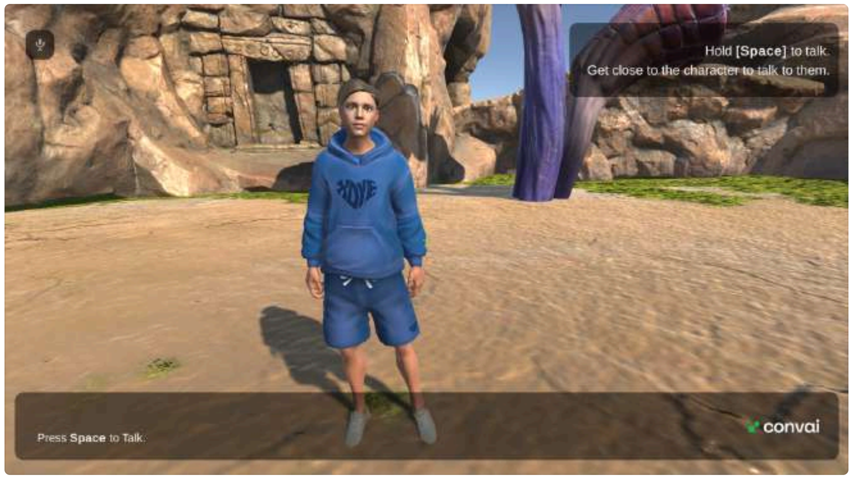


Animator

Controller: Feminine NPC Animator
Avatar: FeminineAvatar
Apply Root Motion: ☐
Update Mode: Normal
Culling Mode: Always Animate

For female avatars

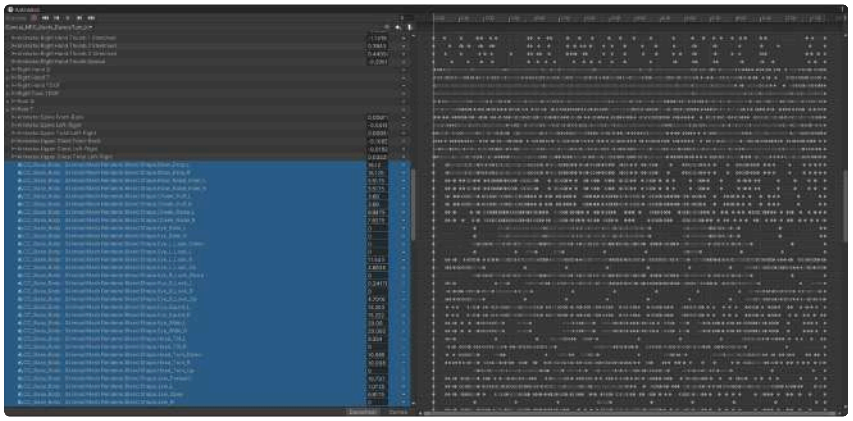
The correct animation will look something like this. The hands should not intersect the body.



Animations have Facial Blendshapes

Resolve facial blendshape issues in Unity animations with Convai. Improve character realism.

If the Lip-sync from characters are either not visible or are very faint, if could be a result of character's animations overriding the blendshape changes made by the script. We recommend deleting the relevant components in the animation dopesheet.



The blendshapes in the CC_Base_Body's Skinned Mesh Renderer. We shall delete these.

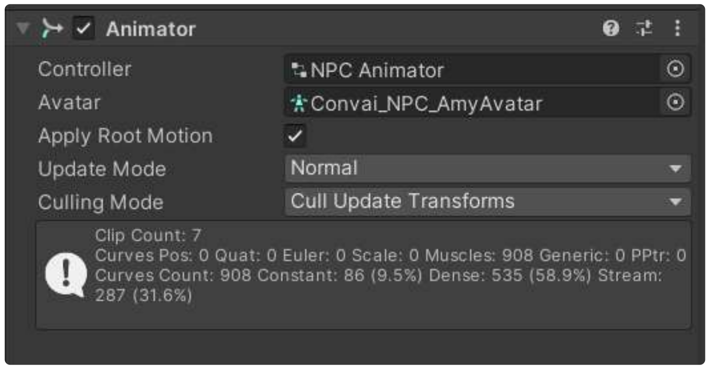
Jaw Bone in Avatar is not Free

Fix jaw bone issues in Unity avatars with Convai. Ensure smooth lip sync and animations.

If the Lip Sync does not seem to cause any facial animations, even after removing all blendshapes from animations, then the following steps should help resolve the issue.

ⓘ This is a known issue in Reallusion CC4 characters.

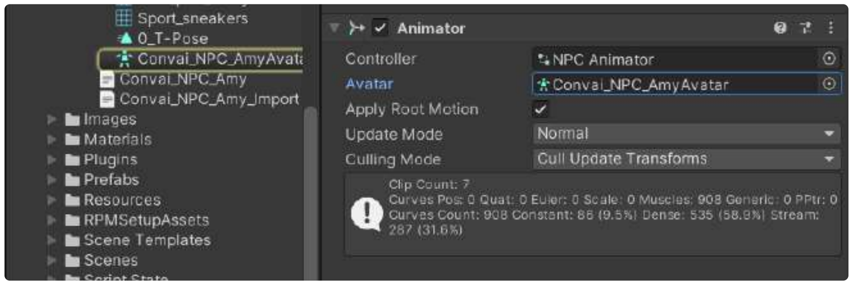
Select the Character and head to the Animator component.



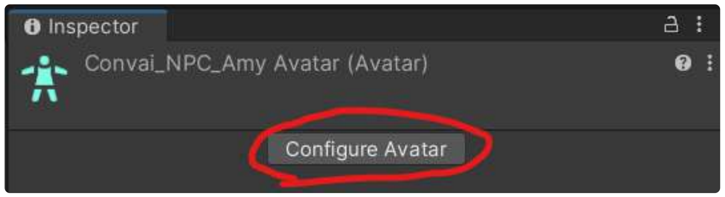
Controller: NPC Animator
Avatar: Convai_NPC_AmyAvatar
Apply Root Motion: ✓
Update Mode: Normal
Culling Mode: Cull Update Transforms

```
Clip Count: 7
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 908 Generic: 0 PPtr: 0
Curves Count: 908 Constant: 86 (9.5%) Dense: 535 (58.9%) Stream:
287 (31.6%)
```

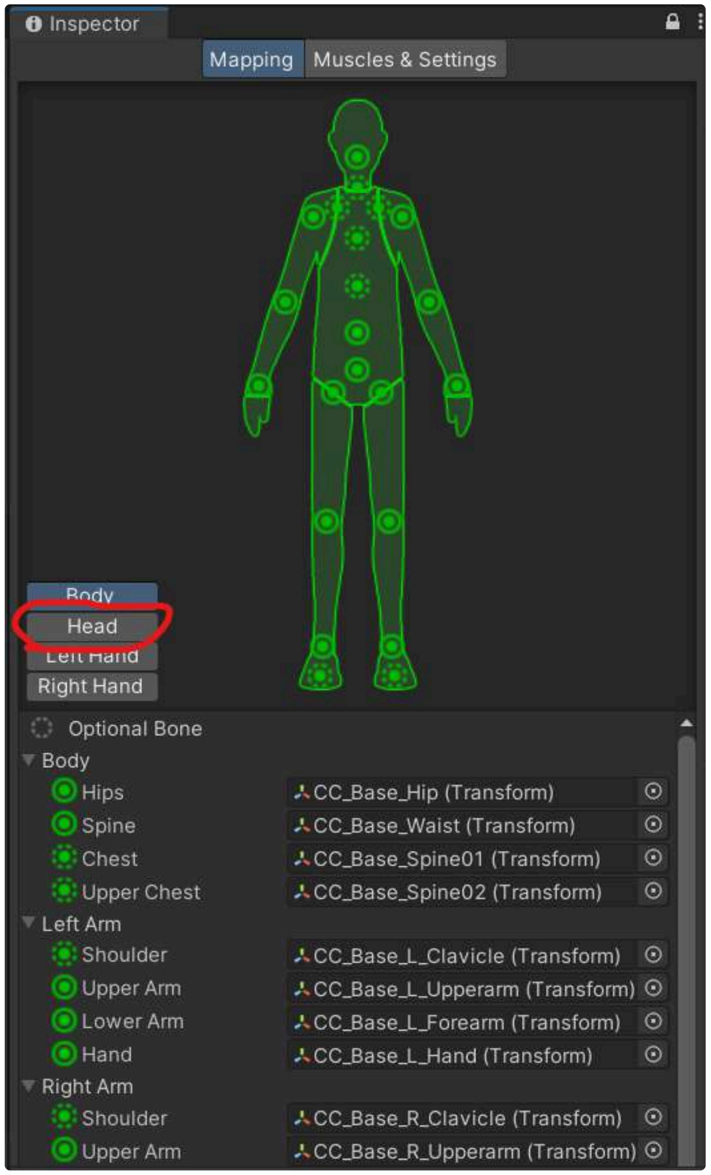
Click the Avatar Field once to select the character's avatar in the Project window.



Select the Avatar and click Configure Avatar.



Select the Head option in the Mapping tab.



Body

Hips	CC_Base_Hip (Transform)
Spine	CC_Base_Waist (Transform)
Chest	CC_Base_Spine01 (Transform)
Upper Chest	CC_Base_Spine02 (Transform)

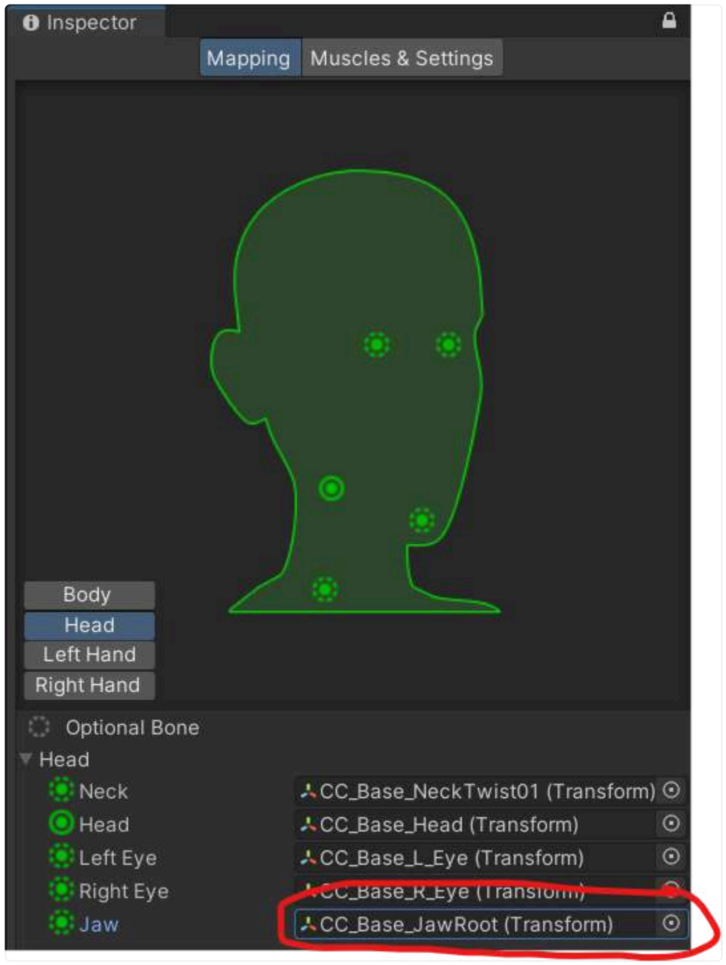
Left Arm

Shoulder	CC_Base_L_Clavicle (Transform)
Upper Arm	CC_Base_L_Upperarm (Transform)
Lower Arm	CC_Base_L_Forearm (Transform)
Hand	CC_Base_L_Hand (Transform)

Right Arm

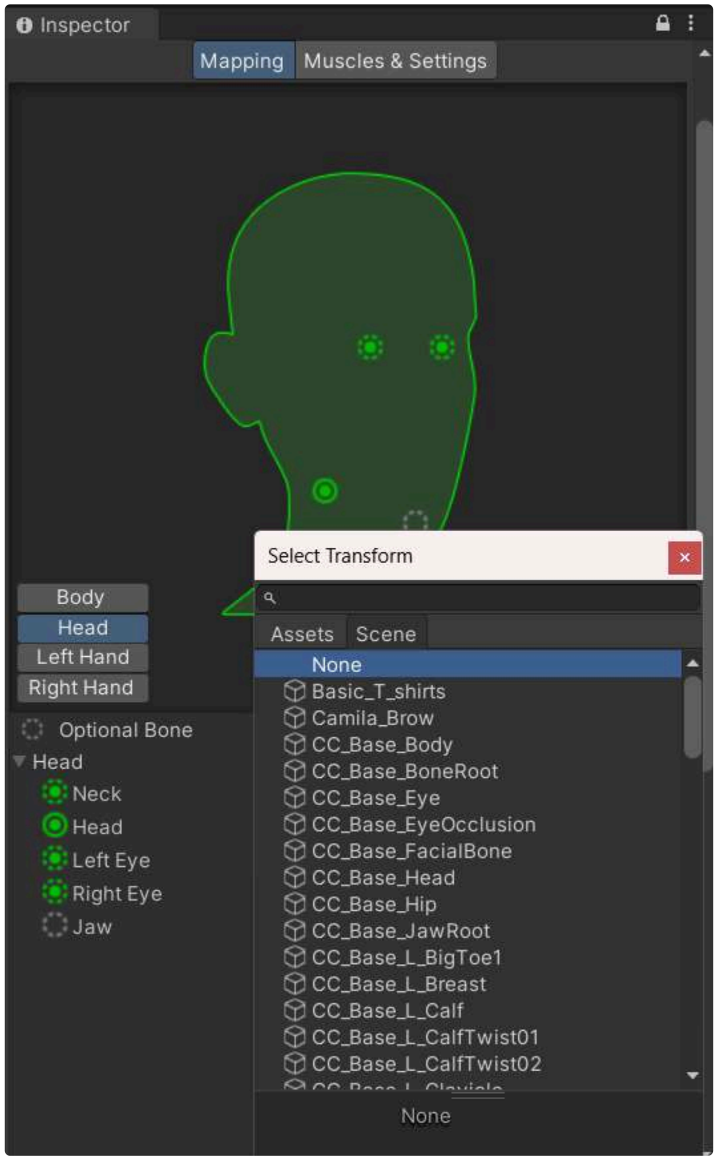
Shoulder	CC_Base_R_Clavicle (Transform)
Upper Arm	CC_Base_R_Upperarm (Transform)

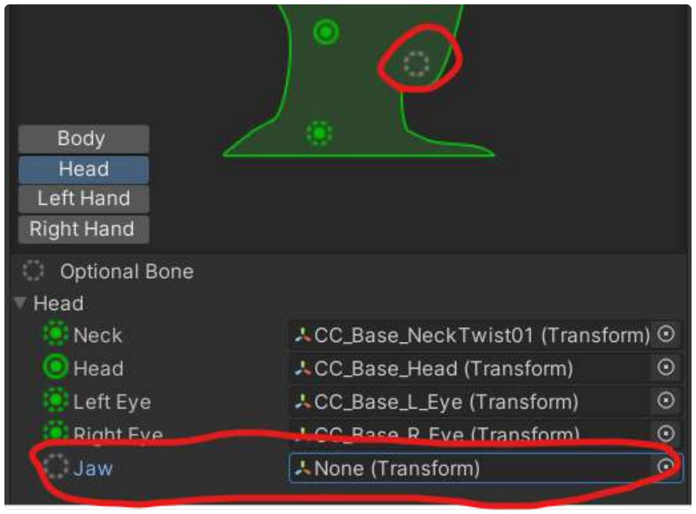
Select the Jaw Mapping and set it to None.



Head

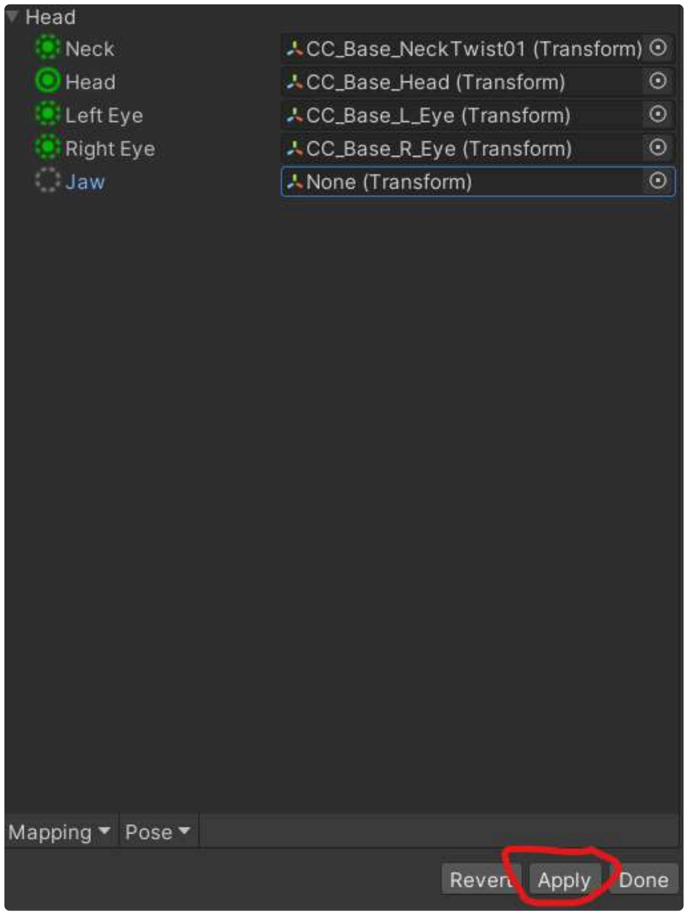
Neck	CC_Base_NeckTwist01 (Transform)
Head	CC_Base_Head (Transform)
Left Eye	CC_Base_L_Eye (Transform)
Right Eye	CC_Base_R_Eye (Transform)
Jaw	CC_Base_JawRoot (Transform)





Neck	CC_Base_NeckTwist01 (Transform)
Head	CC_Base_Head (Transform)
Left Eye	CC_Base_L_Eye (Transform)
Right Eye	CC_Base_R_Eye (Transform)
Jaw	None (Transform)

Finally scroll down and click Apply.


Head

Neck	CC_Base_NeckTwist01 (Transform)
Head	CC_Base_Head (Transform)
Left Eye	CC_Base_L_Eye (Transform)
Right Eye	CC_Base_R_Eye (Transform)
Jaw	None (Transform)

This will free the avatar's jaw mapping and allow the script to manipulate the Jaw bones.

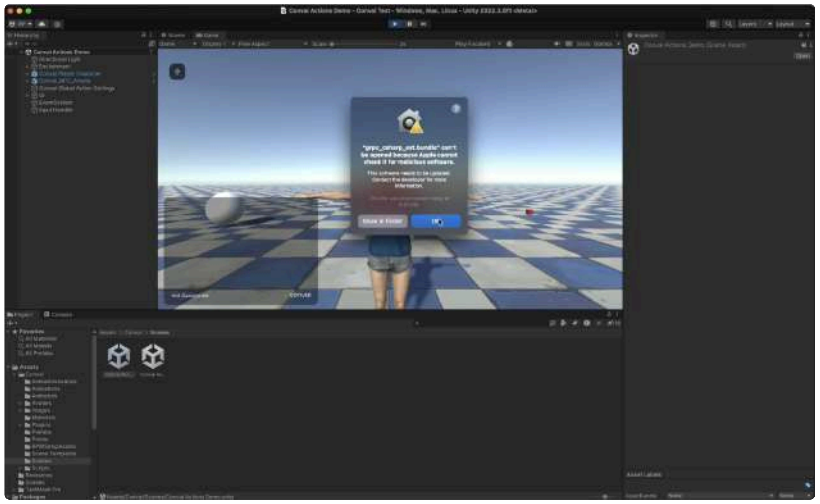
macOS Permission Issues

macOS security permission issue with custom DLLs in Unity and Mac Configuration in build settings

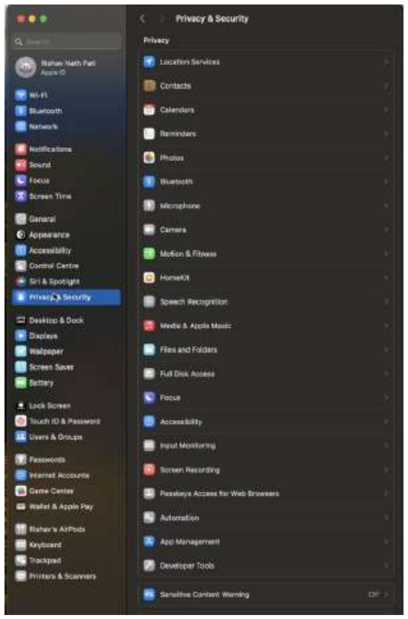
Allowing the grpc_csharp_ext.bundle dll file in macOS

Using external DLLs in Unity on MacOS can lead to security permission issues due to Apple's strict security measures. Here's a step-by-step guide to resolving this common problem.

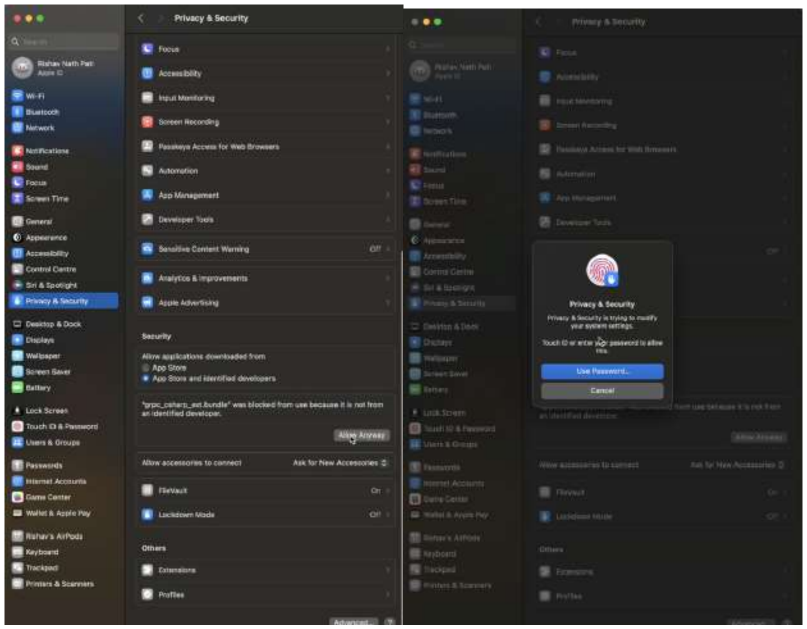
1. **Verify the Problem**:



2. **Manually Allow Blocked DLLs**:
   - Open System Preferences on your Mac.
   - Navigate to "Security & Privacy".



- Under the "Security" tab, you might see a message at the bottom about the DLL being blocked. Click "Allow Anyway" or "Open Anyway" and enter password if asked.


Privacy & Security
Security
Allow applications downloaded from
App Store
App Store and identified developers
Allow Anyway
Allow accessories to connect Ask for New Accessories
FileVault On
Lockdown Mode Off
Others
Extensions
Profiles
Use Password...
Cancel

"grpc_csharp_ext.bundle" was blocked from use because it is not from an identified developer.

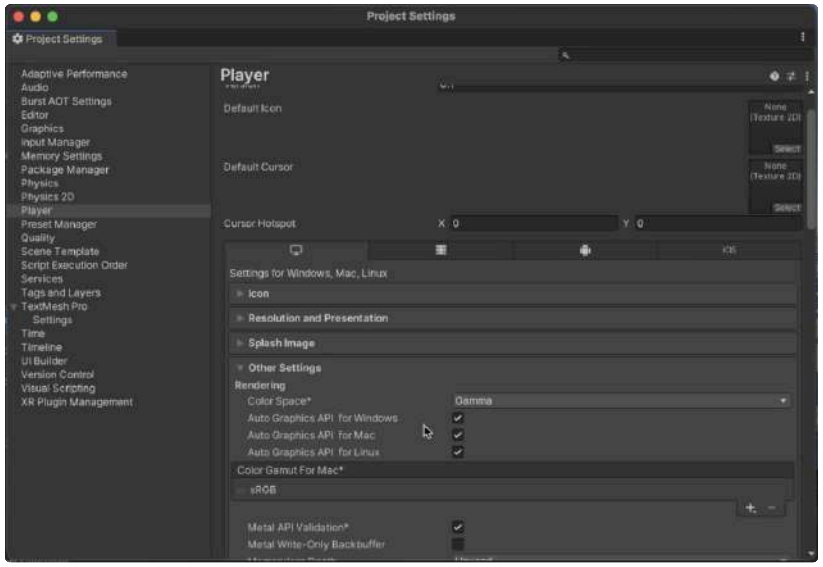
3. **Modify Gatekeeper settings**: MacOS's Gatekeeper can prevent unidentified developers' software from running. To allow the DLL:
   - Open the Terminal (found in Applications > Utilities).
   - Type `sudo spctl --master-disable` and press Enter.
   - This command will allow apps to be downloaded from anywhere.
   - Now, try running the Unity project again.
   - After you're done, you should re-enable Gatekeeper with `sudo spctl --master-enable` to avoid any malware.
4. **Check File Permissions**: Ensure the DLL has the correct file permissions.
   - In Finder, right-click (or control-click) on the DLL file and choose "Get Info".
   - Under “Sharing & Permissions”, ensure that your user account has "Read & Write" permissions.
5. **Review Unity's Plugin Settings**:
   - In the Unity editor, select the DLL in the Project view.
   -

In the Inspector window, make sure the appropriate platform (in this case, Mac OS X) and architecture (Apple Silicon, Intel-64) is selected for the DLL.

- Ensure that the "Load on Startup" and other pertinent options are checked (should be enabled by default)

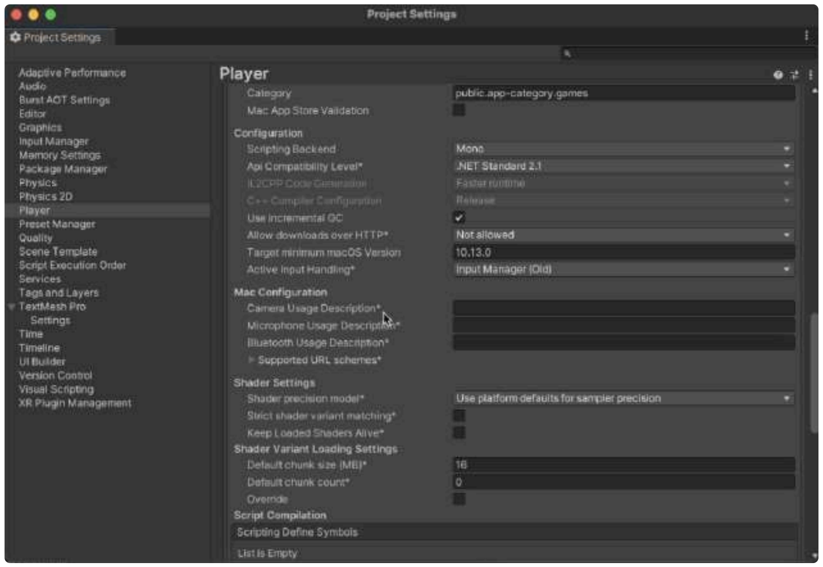
Mac Configuration in Player Settings during build

- **Update Mac Configuration:**
  - In Unity, navigate to `Edit > Project Settings > Player`.
  - Scroll down and click on `Other Settings`



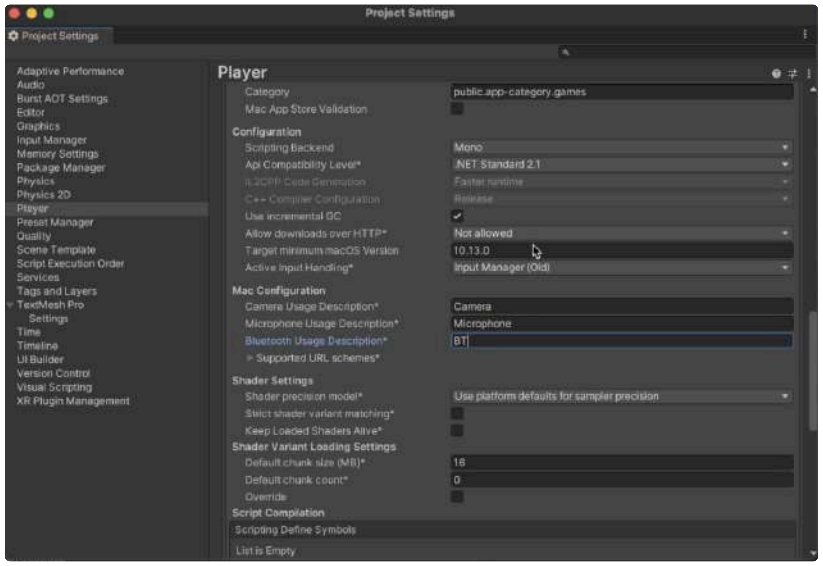
Default Icon: None (Texture 2D)
Default Cursor: None (Texture 2D)
Cursor Hotspot: X 0 Y 0
Settings for Windows, Mac, Linux
Other Settings
Rendering
Color Space*: Gamma
Auto Graphics API for Windows: ✓
Auto Graphics API for Mac: ✓
Auto Graphics API for Linux: ✓
Color Gamut For Mac*: sRGB
Metal API Validation*: ✓
Metal Write-Only Backbuffer: ☐

  - Scroll down again to find Mac Configuration section



Category: public.app-category.games
Mac App Store Validation: ☐
Configuration
Scripting Backend: Mono
Api Compatibility Level*: .NET Standard 2.1
IL2CPP Code Generation: Faster runtime
C++ Compiler Configuration: Release
Use incremental GC: ✓
Allow downloads over HTTP*: Not allowed
Target minimum macOS Version: 10.13.0
Active Input Handling*: Input Manager (Old)
Mac Configuration
Camera Usage Description*:
Microphone Usage Description*:
Bluetooth Usage Description*:
Supported URL schemes*
Shader Settings
Shader precision model*: Use platform defaults for sampler precision
Strict shader variant matching*: ☐
Keep Loaded Shaders Alive*: ☐
Shader Variant Loading Settings
Default chunk size (MB)*: 16
Default chunk count*: 0
Override: ☐
Script Compilation
Scripting Define Symbols
List is Empty

  - Update the Mac Configuration section (follow the below Screenshot)



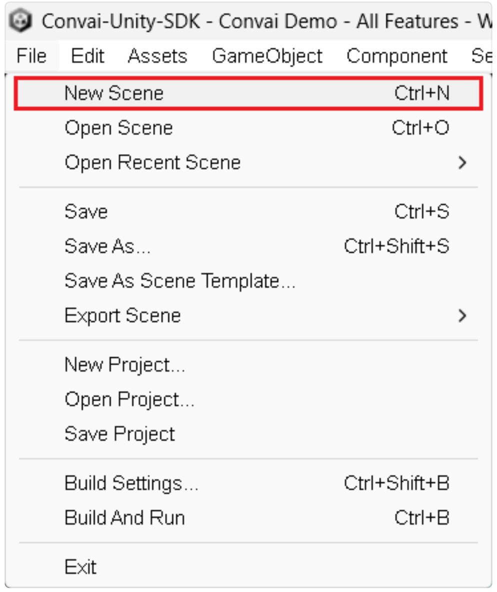
Category: public.app-category.games
Mac App Store Validation: ☐
Configuration
Scripting Backend: Mono
Api Compatibility Level*: .NET Standard 2.1
IL2CPP Code Generation: Faster runtime
C++ Compiler Configuration: Release
Use incremental GC: ✓
Allow downloads over HTTP*: Not allowed
Target minimum macOS Version: 10.13.0
Active Input Handling*: Input Manager (Old)
Mac Configuration
Camera Usage Description*: Camera
Microphone Usage Description*: Microphone
Bluetooth Usage Description*: BT
Supported URL schemes*
Shader Settings
Shader precision model*: Use platform defaults for sampler precision
Strict shader variant matching*: ☐
Keep Loaded Shaders Alive*: ☐
Shader Variant Loading Settings
Default chunk size (MB)*: 16
Default chunk count*: 0
Override: ☐
Script Compilation
Scripting Define Symbols
List is Empty

Creating a Convai Powered Scene from Template

This guide will help you make a scene in unity with Convai Essentials already present in it. It will help you to get started with our plugin very fast.

Step 1) Open the *New Scene* window

You can open the new scene window by two ways, first by pressing `Ctrl + N` for windows or `CMD + N` for Mac on your keyboard, second way is to navigate to File -> New Scene

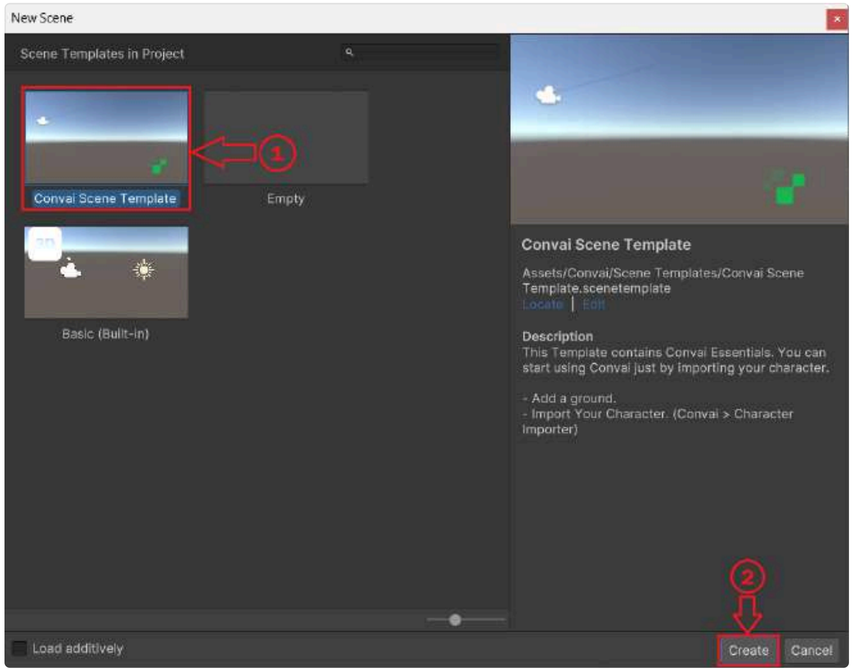


Screenshot showing process of opening up New Scene Window in Unity

Step 2) Select Convai Scene Template

There will be many scene templates depending upon your project, but in this guide, we are interested in Convai Scene Template so select that and click on Create

button.



Convai Scene Template

Assets/Convai/Scene Templates/Convai Scene Template.scenetemplate

Locate | Edit

Description

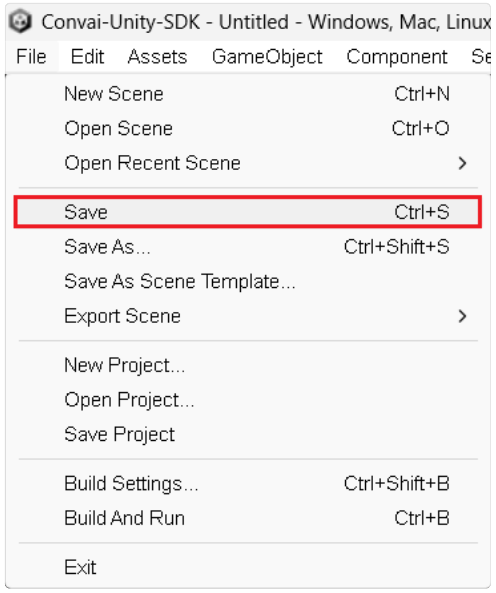
This Template contains Convai Essentials. You can start using Convai just by importing your character.

- Add a ground.
- Import Your Character. (Convai > Character Importer)

Screenshot showing how to create a scene from convai scene template

Step 3) Save Created Scene

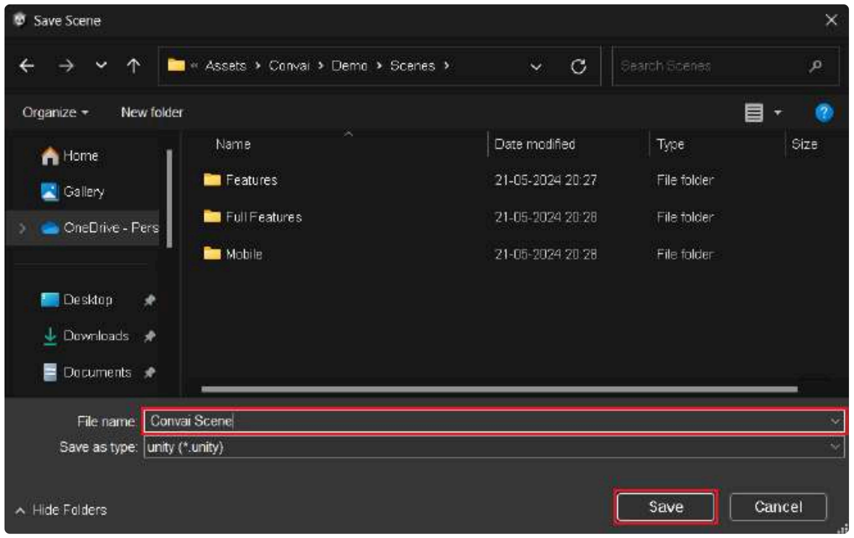
You can now save the newly created scene in the project at your desired location by either pressing `Ctrl + S` on Windows or `CMD + S` on Mac. Another method is to navigate to File → Save Scene



Screenshot showing how to save the scene

This is open the Save Scene Window, choose your desired location, for this demo we will save it inside Demo folder, but you can save it anywhere in the assets directory.

Give your scene a name and then click on Save Scene button



Name	Date modified	Type	Size
Features	21-05-2024 20:27	File folder	
Full Features	21-05-2024 20:28	File folder	
Mobile	21-05-2024 20:28	File folder	

File name: Convai Scene
Save as type: unity (*.unity)

Screenshot showing save location of the new Convai powered scene

Now you can import your Convai Character or your Custom Characters by following our complete guide on it

Importing RPM Characters

Importing Custom Characters

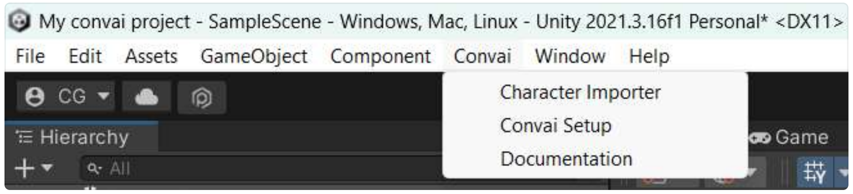
Importing RPM Characters

Follow these instructions to bring a character from the Convai Playground into your Unity Project.

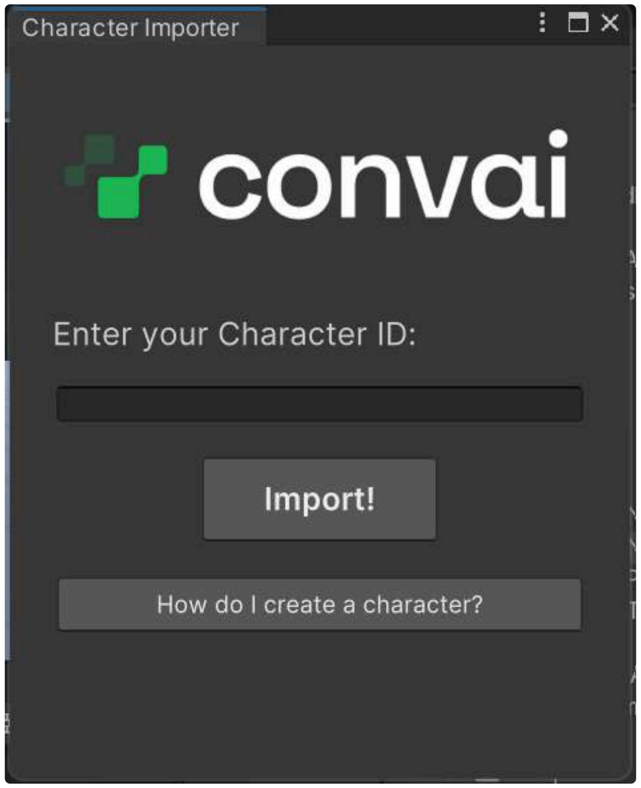
Import RPM characters from the Convai Playground

This is how you can import characters from the Convai Playground into your Unity Project.

In the Menu Bar, go the Convai > Character Importer.


My convai project - SampleScene - Windows, Mac, Linux - Unity 2021.3.16f1 Personal* <DX11>
File Edit Assets GameObject Component Convai Window Help
Character Importer
Convai Setup
Documentation
CG
Hierarchy
All
Game

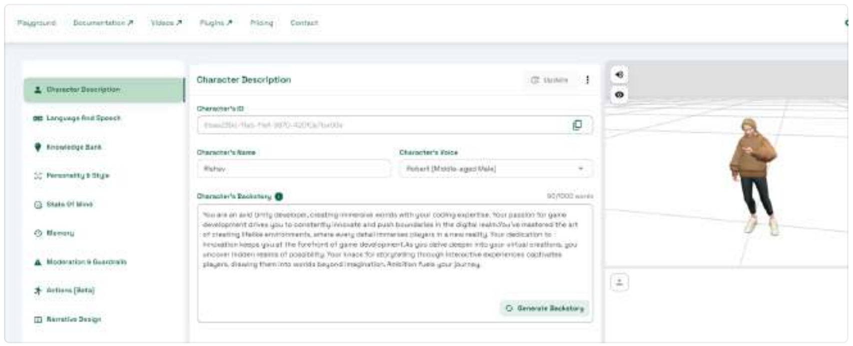
Enter the Character ID and click Import.



Enter the character ID and click Import.

If you are unsure how to get the character ID, click the "How do I create a character?".

You can get the character ID from the Character Description.



Character Description

Character's ID: 6bae239c-11ab-11ef-9870-42010a7be00e

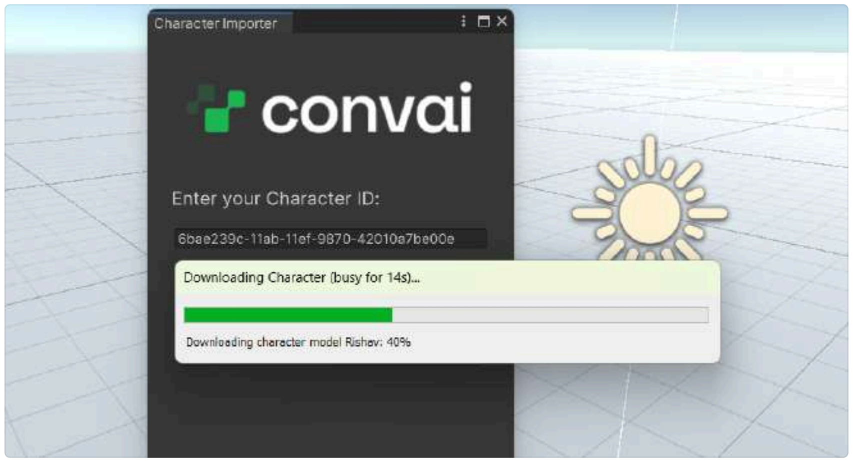
Character's Name: Rishav

Character's Voice: Robert (Middle-aged Male)

Character's Backstory 90/1000 words

You are an avid Unity developer, creating immersive worlds with your coding expertise. Your passion for game development drives you to constantly innovate and push boundaries in the digital realm.You've mastered the art of creating lifelike environments, where every detail immerses players in a new reality. Your dedication to innovation keeps you at the forefront of game development.As you delve deeper into your virtual creations, you uncover hidden realms of possibility. Your knack for storytelling through interactive experiences captivates players, drawing them into worlds beyond imagination. Ambition fuels your journey.

The downloading will take a while. On successful download, you will see the character in the scene with the same GameObject as the character ID.

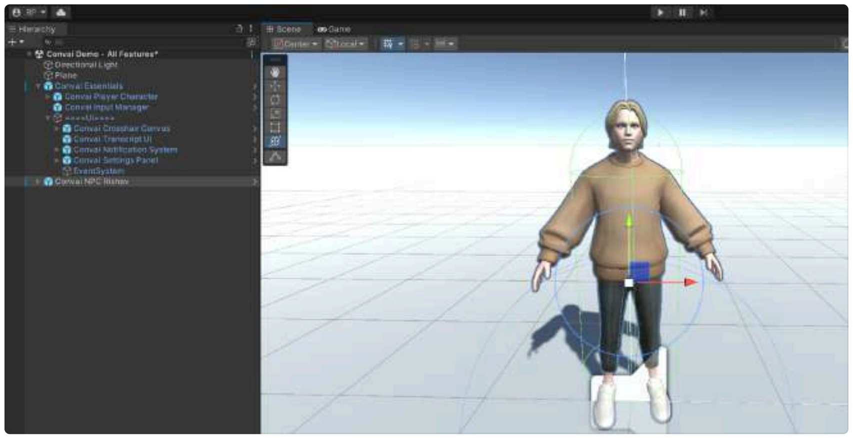


Enter your Character ID:

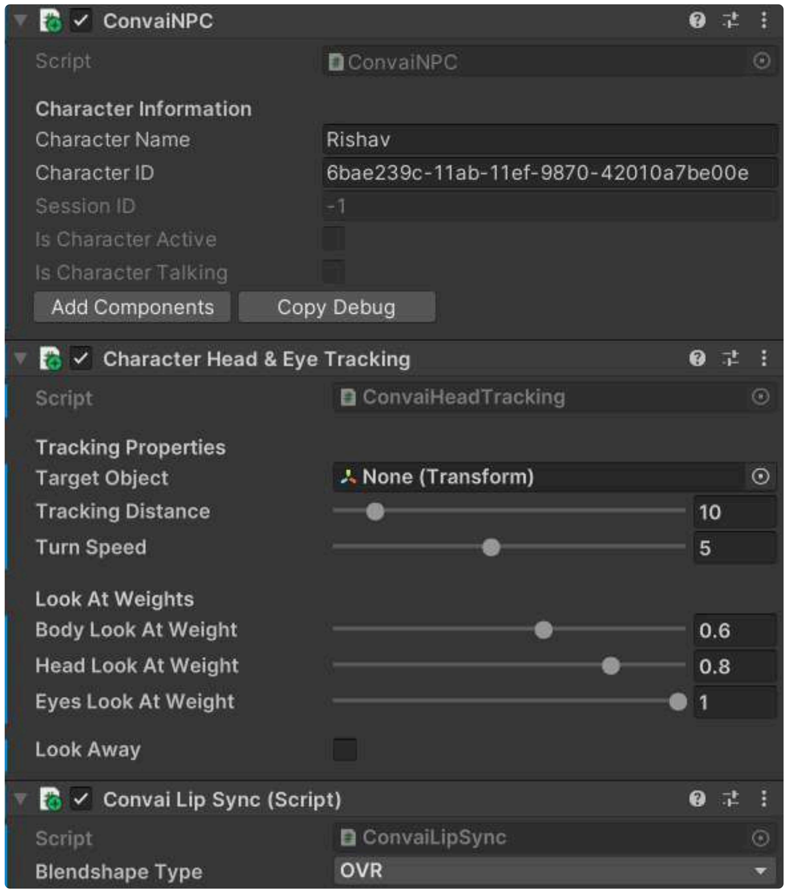
6bae239c-11ab-11ef-9870-42010a7be00e

Downloading Character (busy for 14s)...

Downloading character model Rishav: 40%



This character will automatically be set up with the basic Convai Setup including the ConvaiNPC Script and Out-Of-Box Animations.

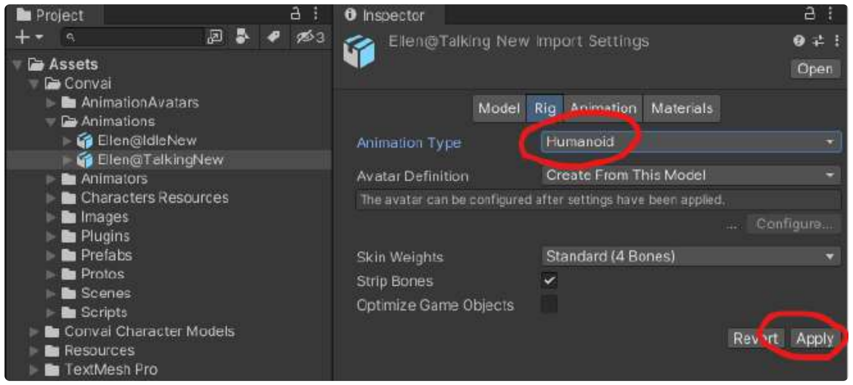


ConvaiNPC
Script: ConvaiNPC
Character Information
Character Name: Rishav
Character ID: 6bae239c-11ab-11ef-9870-42010a7be00e
Session ID: -1
Is Character Active
Is Character Talking
Add Components | Copy Debug

Character Head & Eye Tracking
Script: ConvaiHeadTracking
Tracking Properties
Target Object: None (Transform)
Tracking Distance: 10
Turn Speed: 5
Look At Weights
Body Look At Weight: 0.6
Head Look At Weight: 0.8
Eyes Look At Weight: 1
Look Away

Convai Lip Sync (Script)
Script: ConvaiLipSync
Blendshape Type: OVR

If you are facing issues with the animations in your imported character, make sure to change the animation type of `Ellen@IdleNew` and `Ellen@TalkingNew` Animations in the `Assets/Convai/Animations` folder to `Humanoid`.



Project
- Assets
  - Convai
    - AnimationAvatars
    - Animations
      - Ellen@IdleNew
      - Ellen@TalkingNew
    - Animators
    - Characters Resources
    - Images
    - Plugins
    - Prefabs
    - Protos
    - Scenes
    - Scripts
  - Convai Character Models
  - Resources
  - TextMesh Pro

Inspector
Ellen@Talking New Import Settings
Open
Model | Rig | Animation | Materials
Animation Type: Humanoid
Avatar Definition: Create From This Model
The avatar can be configured after settings have been applied.
Configure...
Skin Weights: Standard (4 Bones)
Strip Bones: ✓
Optimize Game Objects
Revert | Apply

Change the animation type to 'Humanoid' and click 'Apply'.

Now you are ready to set up the character with transcriptions.

Importing Custom Characters

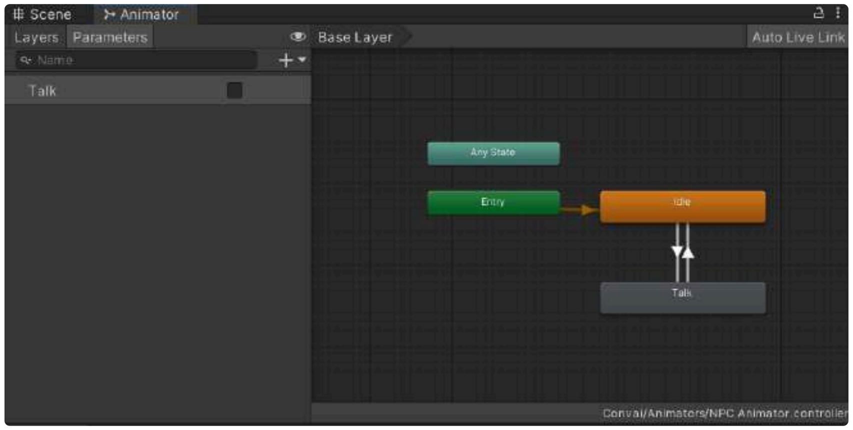
Follow these instructions to set up your imported character with Custom Model with Convai.

To import your custom characters into your Convai-powered Unity project, you will first need to bring your model into your project. The model needs at least two animations: one for talking and one for Idle.

Prerequisites

When you want to set up your custom character with Convai, you will need your character model and two animations: Idle and Talking.

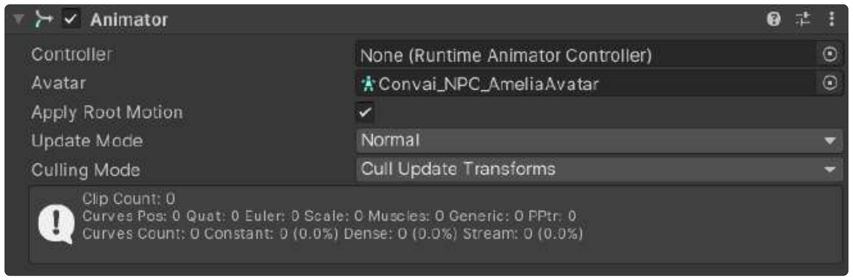
Create an animator controller with the two animations that looks like this. You should also add a 'Talk' Boolean to ensure that you can trigger the animation. Here is a YouTube tutorial on how to set up an animator controller. This is the bare minimum animator setup that you need to do.



The animator controller should look like this. This is the in-box NPC Animator.

Step 1: Add Animator to your custom character

Select your character from the Hierarchy and Add Animator Component



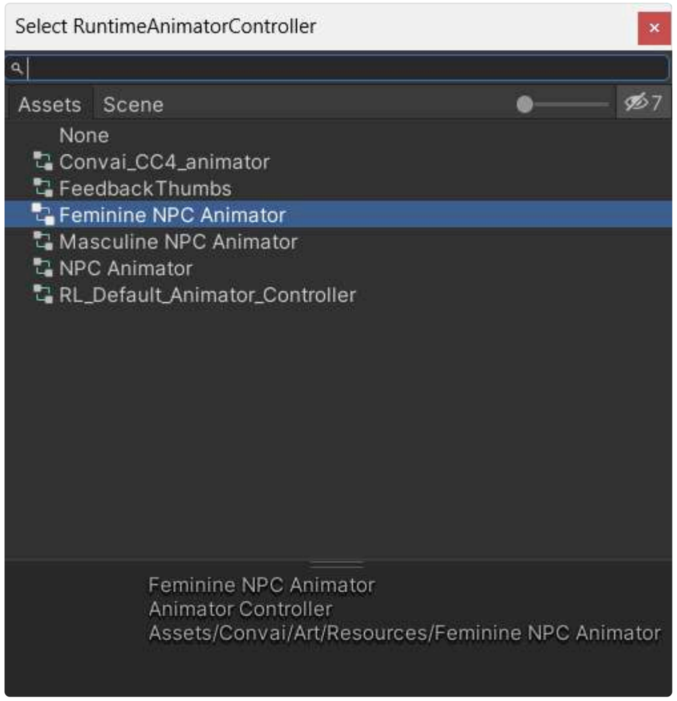
Animator

Controller: None (Runtime Animator Controller)
Avatar: Convai_NPC_AmeliaAvatar
Apply Root Motion: ✓
Update Mode: Normal
Culling Mode: Cull Update Transforms

Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%) Dense: 0 (0.0%) Stream: 0 (0.0%)

Screenshot showing newly added Animator Component

Convai Plugin ships with two pre-made animation controller, you can choose these controllers or can assign your custom controller, whatever fits your need. For this demo we are going with `Feminine NPC Animator`



Select RuntimeAnimatorController

Assets Scene 7

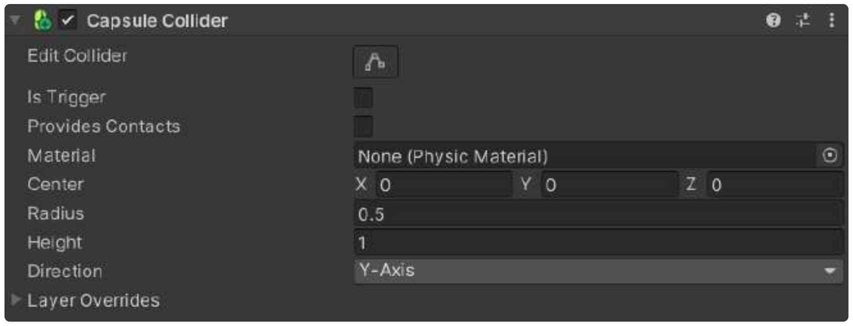
None
Convai_CC4_animator
FeedbackThumbs
Feminine NPC Animator
Masculine NPC Animator
NPC Animator
RL_Default_Animator_Controller

Feminine NPC Animator
Animator Controller
Assets/Convai/Art/Resources/Feminine NPC Animator

Screenshot showing selection of Animation Controller

Step 2: Adding a Trigger Volume

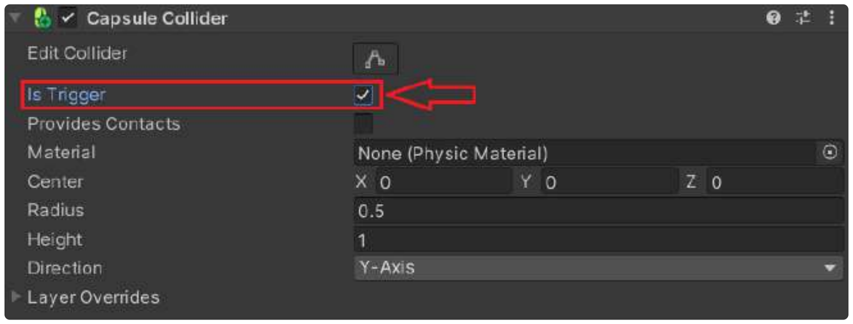
With your custom character selected, add a Collision shape of your choice, for this demo we are going with a `Capsule Collider`



Capsule Collider
Edit Collider
Is Trigger
Provides Contacts
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Radius: 0.5
Height: 1
Direction: Y-Axis
Layer Overrides

Screenshot showing newly added Capsule Collider

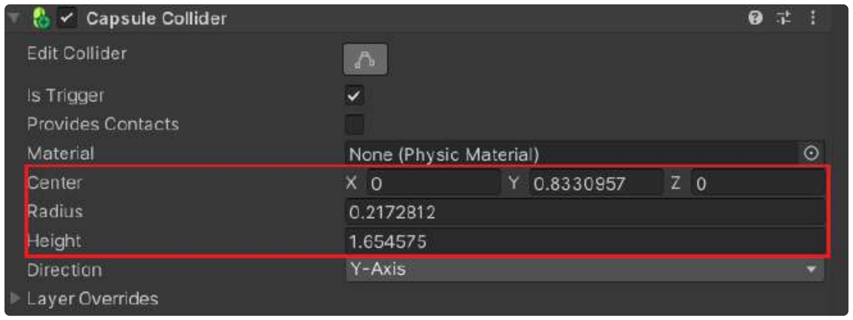
We will make this Collider a trigger, for this we will enable the `Is Trigger` option in the inspector panel



Capsule Collider
Edit Collider
Is Trigger: ✓
Provides Contacts
Material: None (Physic Material)
Center: X 0 Y 0 Z 0
Radius: 0.5
Height: 1
Direction: Y-Axis
Layer Overrides

Screenshot showing enable of Is Trigger option

We will adjust the Center, Radius and Height of the collider such that it fits our character

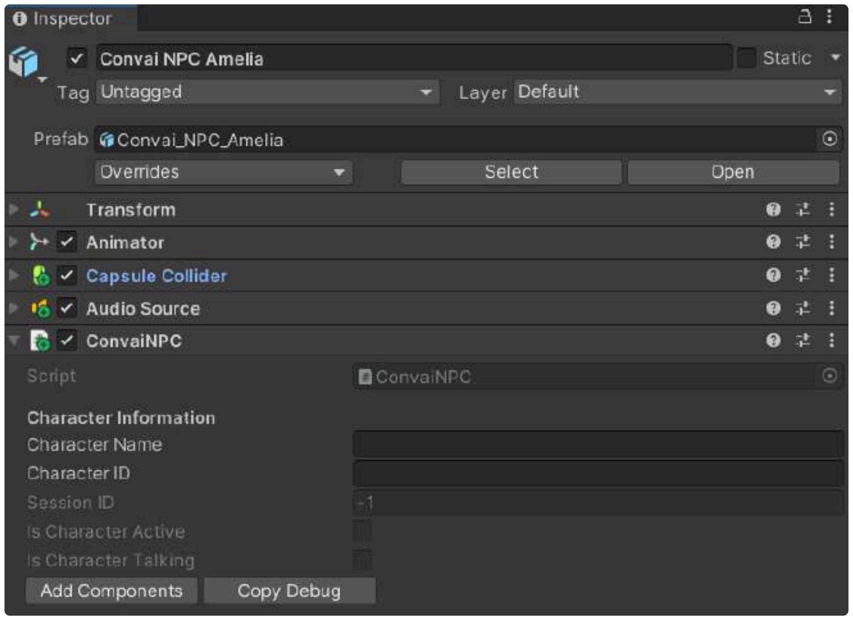


Capsule Collider
Edit Collider
Is Trigger: ✓
Provides Contacts
Material: None (Physic Material)
Center: X 0 Y 0.8330957 Z 0
Radius: 0.2172812
Height: 1.654575
Direction: Y-Axis
Layer Overrides

Step 3: Add ConvaiNPC Component

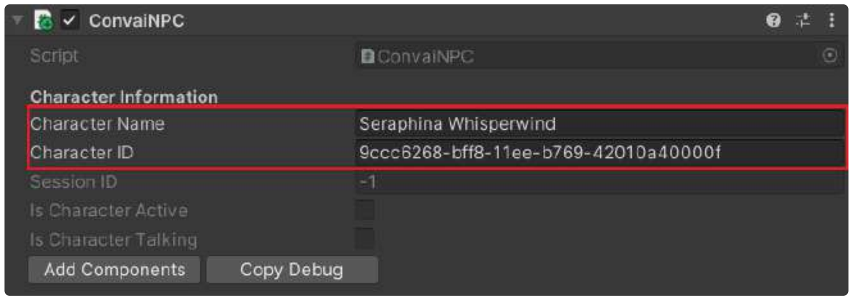
With your Custom Character Selection add ConvaiNPC component. By doing so, your Game objectgame should look like this:

> We assume that nothing other than pre-instructed components were added by you; your Game Object component list may be different



Screenshot showing newly added ConvaiNPC Component

Copy your character's ID and name from Convai Playground and paste them here.

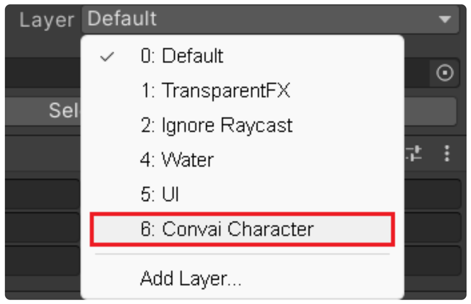


Character Name: Seraphina Whisperwind
Character ID: 9ccc6268-bff8-11ee-b769-42010a40000f

Screenshot showing filled-in character information.

Step 4: Setup a: game object for Convai Character

We will assign `Convai Character` layer to your Custom Character Game Object



Screenshot showing selection of Convai Character

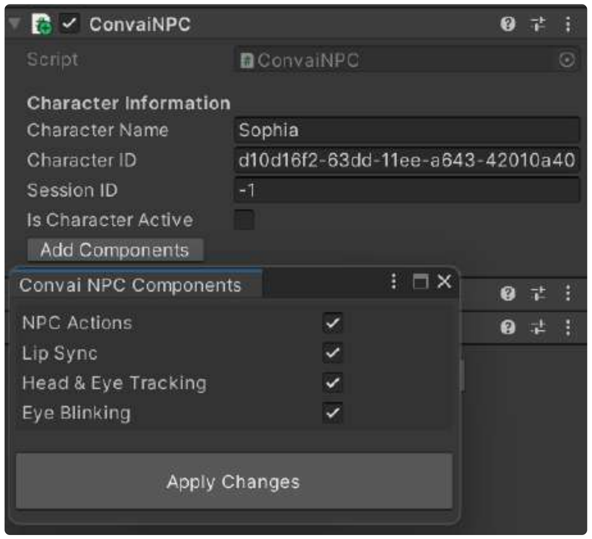
Now your Custom Character is all set to work with Convai Plugin

Adding Actions to your Character

Follow these instructions to enable actions for your Convai-powered characters.

Setting Up Action Configurations

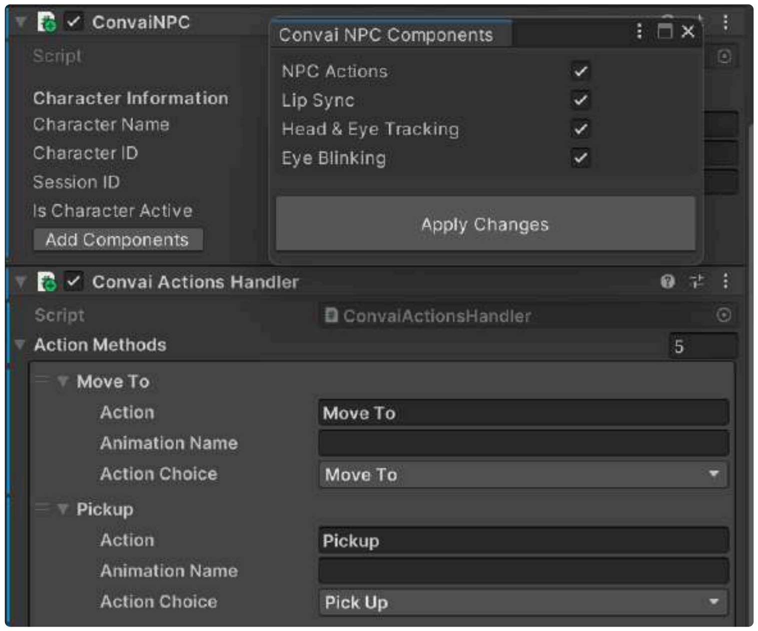
1. Select the Convai NPC character from the hierarchy.
2. Scroll down to the ConvaiNPC script attached to your character.
3. Click the "Add Component" button.



ConvaiNPC
Script: ConvaiNPC
Character Information
Character Name: Sophia
Character ID: d10d16f2-63dd-11ee-a643-42010a40
Session ID: -1
Is Character Active: ☐
Add Components

Convai NPC Components
NPC Actions ✓
Lip Sync ✓
Head & Eye Tracking ✓
Eye Blinking ✓
Apply Changes

4. Use the checkbox to add the action script to the NPC Actions.
5. Click "Apply Changes" to confirm.



ConvaiNPC
Script
Character Information
Character Name
Character ID
Session ID
Is Character Active
Add Components

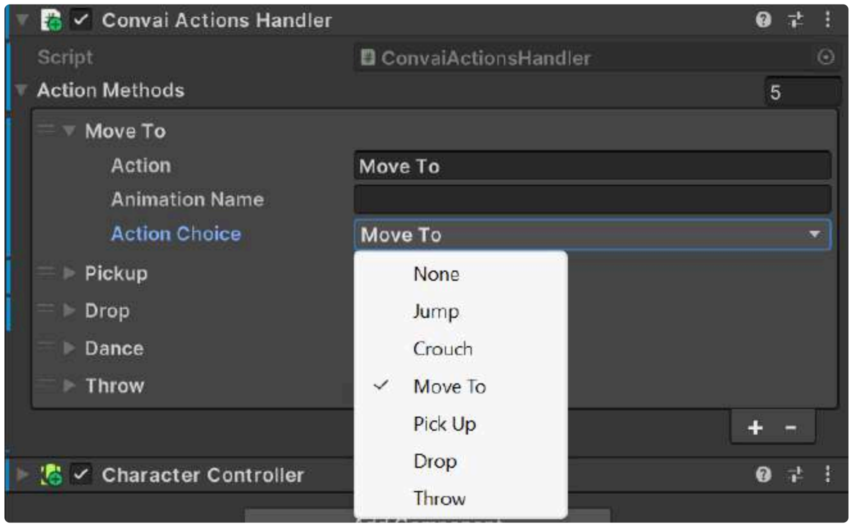
Convai NPC Components
NPC Actions ✓
Lip Sync ✓
Head & Eye Tracking ✓
Eye Blinking ✓
Apply Changes

Convai Actions Handler
Script: ConvaiActionsHandler
Action Methods: 5
Move To
Action: Move To
Animation Name:
Action Choice: Move To
Pickup
Action: Pickup
Animation Name:
Action Choice: Pick Up

Pre-defined Actions

Convai offers predefined actions for a quick start.

1. Click the "+" button to add a new action.
2. From the dropdown menu, select "Move To."



Convai Actions Handler

Script: ConvaiActionsHandler

Action Methods: 5

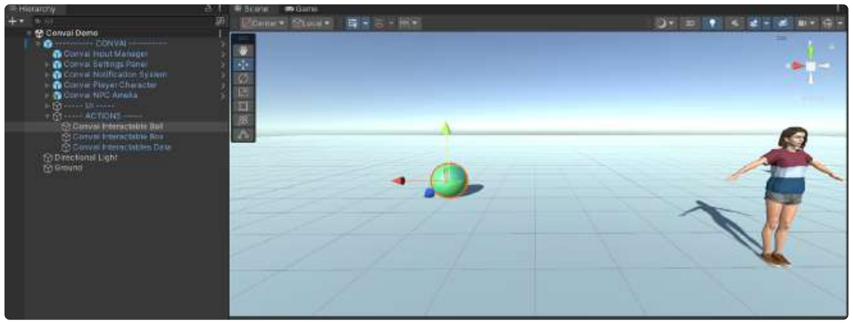
- Move To
  - Action: Move To
  - Animation Name:
  - Action Choice: Move To
- Pickup
- Drop
- Dance
- Throw

3. Enter the action name as "Move To" (the name doesn't have to match the action choice name).
4. Leave the Animation Name field empty for now.

Repeat these steps to add more actions like "Pickup" and "Drop" etc.

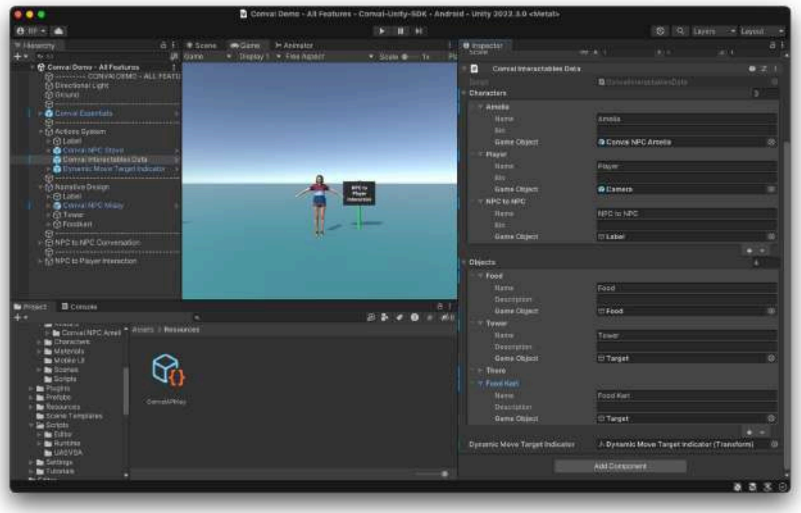
Adding an Object in the Scene

1. Add any object into the scene—a sphere, a cube, a rock, etc.—that can be interacted with
2. Resize and place the object in your scene.



Adding the Convai Interactables Data Script

- Create an empty GameObject and name it "Convai Interactables."
- Attach the Convai Interactables Data script to this GameObject.
- Add characters and objects to the script by clicking the "+" button and attaching the corresponding GameObjects.



Convai Interactables Data

Characters

- Amelia
  - Name: Amelia
  - Bio:
  - Game Object: Convai NPC Amelia
- Player
  - Name: Player
  - Bio:
  - Game Object: Camera
- NPC to NPC
  - Name: NPC to NPC
  - Bio:
  - Game Object: Label

Objects

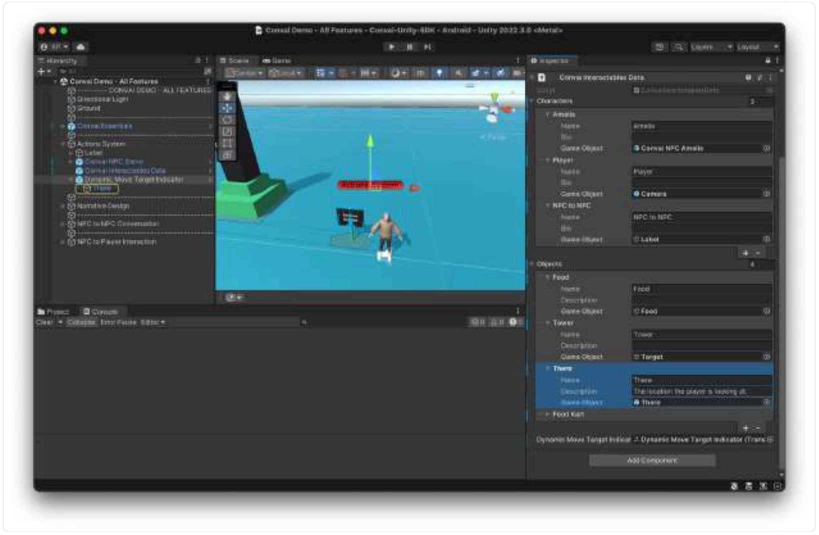
- Food
  - Name: Food
  - Description:
  - Game Object: Food
- Tower
  - Name: Tower
  - Description:
  - Game Object: Target
- There
- Food Kart
  - Name: Food Kart
  - Description:
  - Game Object: Target

Dynamic Move Target Indicator: Dynamic Move Target Indicator (Transform)

Convai Interactables Setup

-

Add the "There" object in Objects list, so that we can use the Dynamic Move Target indicator.

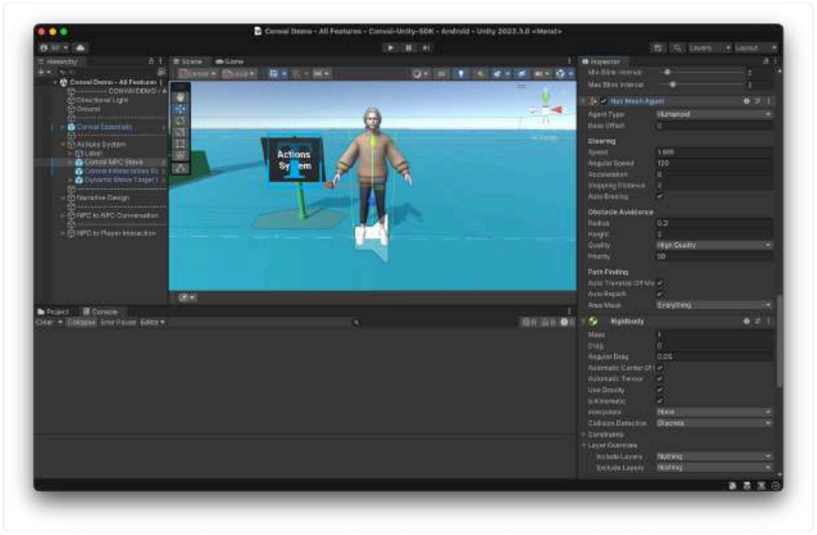


- Add the Dynamic Move Target Indicator and setup NavMesh agent to you NPC.

Setting Up NavMesh

To ensure your NPCs can navigate the scene:

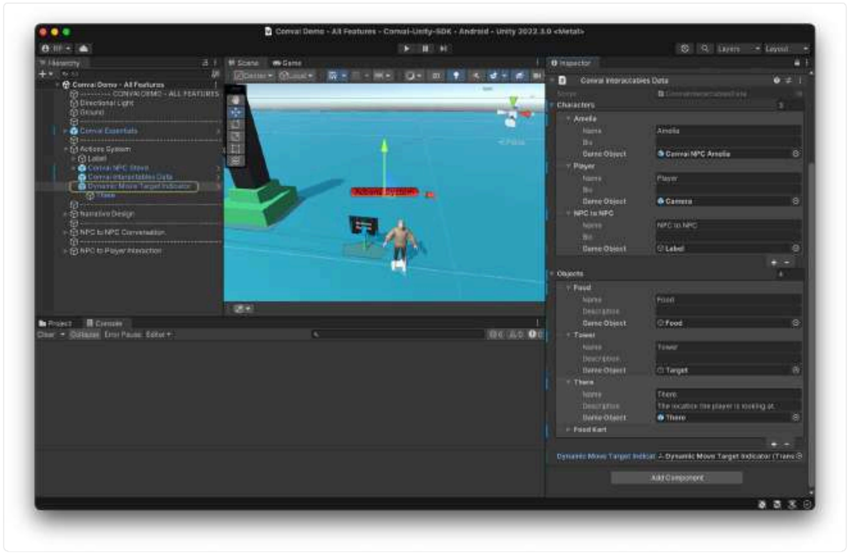
1. **Bake a NavMesh** for your scene if you haven't already:
   - Go to **Window > AI > Navigation**.
   - In the **Navigation** window, under the **Bake** tab, adjust the settings as needed.
   - Click **"Bake"** to generate the NavMesh.
2. Ensure that the NPC character has a **NavMeshAgent** component:
   - If not already attached, click **"Add Component"** and search for **NavMeshAgent**.
   - Adjust the **Agent Radius, Speed,** and other parameters according to your NPC's requirements.



Adding a Dynamic Move Target Indicator

To visually indicate where your NPC will move:

- Create a new empty GameObject in the scene and name it accordingly or use the pre-made prefab named **Dynamic Move Target Indicator.**
- Link this **Move Target Indicator** to your NPC's action script so it updates dynamically when you point the cursor to the ground and ask the NPC to move to "There".



Test the Setup

1. Click "Play" to start the scene.
2. Ask the NPC, "Bring me the Box."
3. If setup properly, the NPC should walk upto the box and bring it to you

ⓘ This feature is currently experimental and can misbehave. Feel free to try it out and leave us any feedback.

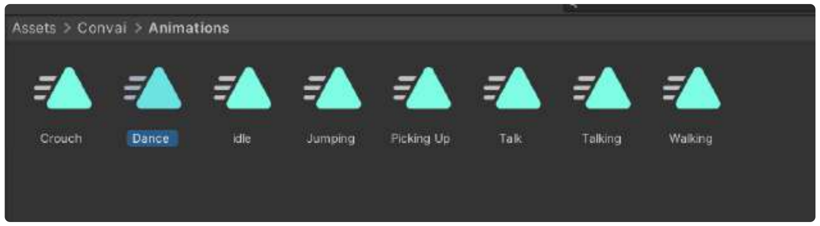
Adding Custom Actions to Your Unity NPC in Convai

Introduction

Make your NPC perform custom actions like dancing.

Action that Only Requires an Animation

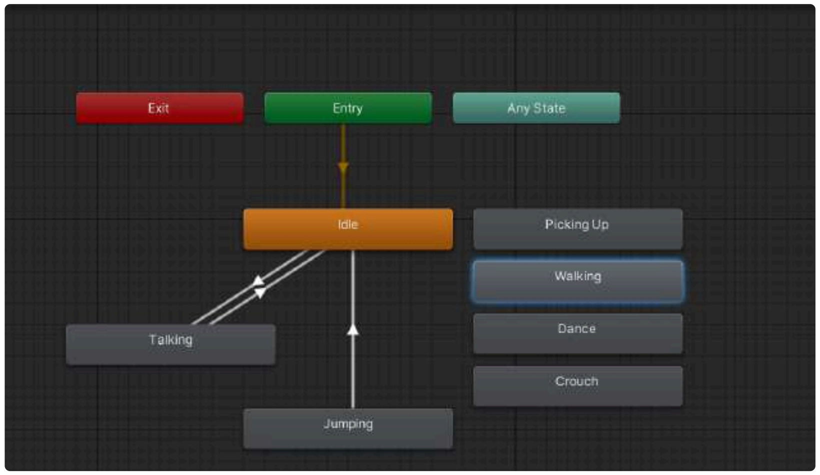
1. Locate the dance animation file within our plugin.



2. Incorporate this animation into your NPC's actions.

Setting Up the Animator Controller

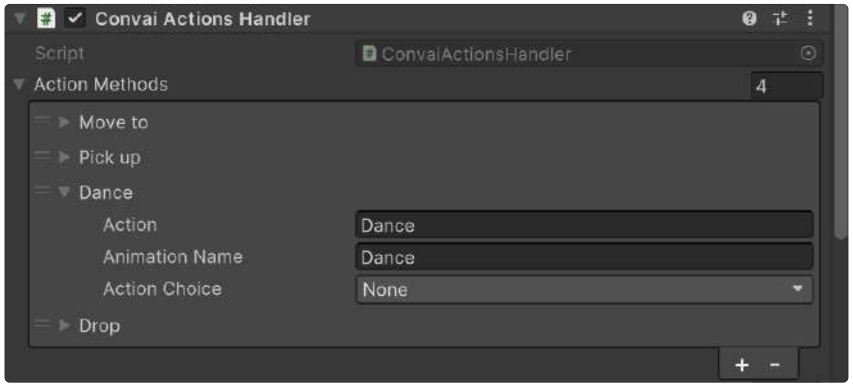
1. Open the Animator Controller from the Inspector window.
2. Drag and drop the dance animation onto the controller, creating a new node named "Dancing."



Adding custom Animation Action

1. Go to the Action Handler Script attached to your Convai NPC.

2. Add a new action named "Dancing."
3. In the Animation Name field, enter "Dancing" (it must exactly match the Animator Controller node name).
4. Leave the enum as "None."


Convai Actions Handler
Script: ConvaiActionsHandler
Action Methods: 4
- Move to
- Pick up
- Dance
  - Action: Dance
  - Animation Name: Dance
  - Action Choice: None
- Drop

Testing the Custom Action

1. Click "Play" to start the scene.
2. Instruct the NPC, "Show me a dance move," and the NPC should start dancing.

Creating Complex Custom Actions in Unity with Convai: Throwing a Rock

Introduction

Adding advanced custom actions, such as a throw action, to your NPC.

Animation Requirement

1. Grab a throw animation from Mixamo or anywhere you like.

2. Import it into Unity.

Setting Up the Animator Controller

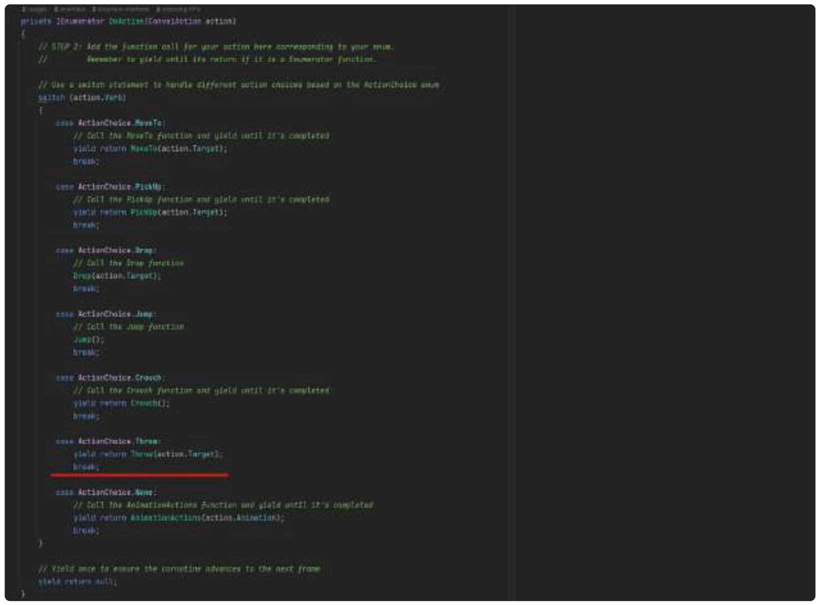
1. Drag and drop the throw animation onto the controller, creating a new node named "Throwing." (Follow steps in #action-that-only-requires-an-animation)

Action Handler Script Setup

1. Add the "Throw" enum to the script.

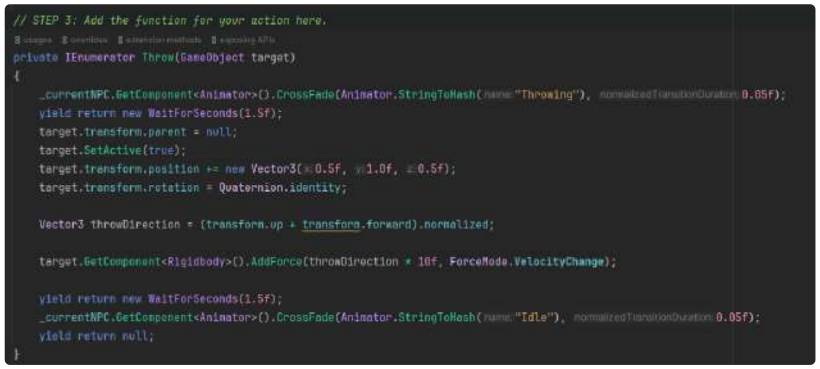
```
// STEP 1: Add the enum for your custom action here.
public enum ActionChoice
{
    None,
    Jump,
    Crouch,
    MoveTo,
    PickUp,
    Drop,
    Throw
}
```

2. In the "Do Action" function, add a switch case for the throw action.



```
private IEnumerator DoAction(ConvaiAction action)
{
    // STEP 2: Add the function call for your action here corresponding to your enum.
    //         Remember to yield until its return if it is a Enumerator function.

    // Use a switch statement to handle different action choices based on the ActionChoice enum
    switch (action.Verb)
    {
        case ActionChoice.MoveTo:
            // Call the MoveTo function and yield until it's completed
            yield return MoveTo(action.Target);
            break;

        case ActionChoice.PickUp:
            // Call the PickUp function and yield until it's completed
            yield return PickUp(action.Target);
            break;

        case ActionChoice.Drop:
            // Call the Drop function
            Drop(action.Target);
            break;

        case ActionChoice.Jump:
            // Call the Jump function
            Jump();
            break;

        case ActionChoice.Crouch:
            // Call the Crouch function and yield until it's completed
            yield return Crouch();
            break;

        case ActionChoice.Throw:
            yield return Throw(action.Target);
            break;

        case ActionChoice.None:
            // Call the AnimationActions function and yield until it's completed
            yield return AnimationActions(action.Animation);
            break;
    }

    // Yield once to ensure the coroutine advances to the next frame
    yield return null;
}
```

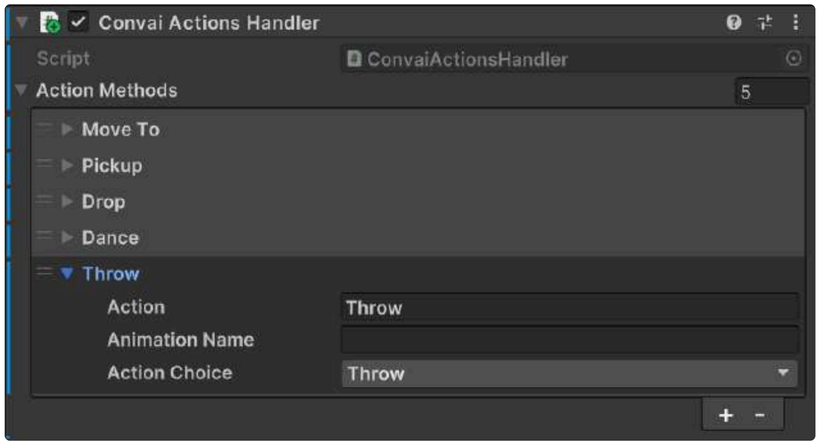
3. Define the "Throw()" function.



```
// STEP 3: Add the function for your action here.
0 usages  0 overrides  0 extension methods  0 exposing APIs
private IEnumerator Throw(GameObject target)
{
    _currentNPC.GetComponent<Animator>().CrossFade(Animator.StringToHash( name: "Throwing"),  normalizedTransitionDuration: 0.05f);
    yield return new WaitForSeconds(1.5f);
    target.transform.parent = null;
    target.SetActive(true);
    target.transform.position += new Vector3( x: 0.5f,  y: 1.0f,  z: 0.5f);
    target.transform.rotation = Quaternion.identity;

    Vector3 throwDirection = (transform.up + transform.forward).normalized;

    target.GetComponent<Rigidbody>().AddForce(throwDirection * 10f, ForceMode.VelocityChange);

    yield return new WaitForSeconds(1.5f);
    _currentNPC.GetComponent<Animator>().CrossFade(Animator.StringToHash( name: "Idle"),  normalizedTransitionDuration: 0.05f);
    yield return null;
}
```

Adding the Throw Action

1. Add a new action named "Throw" and select the "Throw" enum.
2. Leave the animation name field empty.

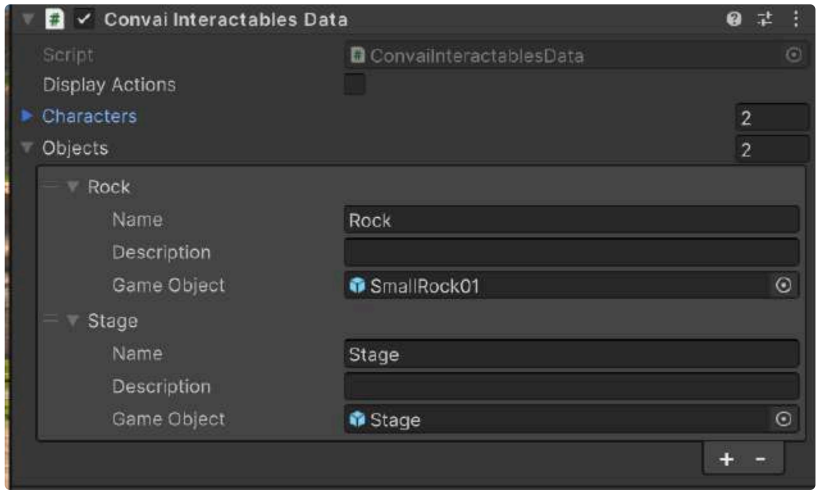


Adding the Object (Rock) to the Convai Interactables Data script

1. Add any rock prefab into the scene.
2. Add the rock to the Convai Interactable Data script.

Adding a location to Convai Interactables Data script

1. Add a stage/new location in the ground of the scene.
2. Add that new location game object in the Convai Interactable Data.



Convai Interactables Data

Script	ConvaiInteractablesData
Display Actions	
Characters	2
Objects	2
Rock	
Name	Rock
Description	
Game Object	SmallRock01
Stage	
Name	Stage
Description	
Game Object	Stage

Testing the Complex Action

1. Click "Play" to start the scene.
2. Instruct the NPC, "Pick up the rock and throw it from the stage."
3. If everything is set up properly, the NPC should pick up the rock and throw it from the stage.

Adding Lip-Sync to your Character

Learn to add lip sync to your Unity characters using Convai. Improve realism and interactivity.

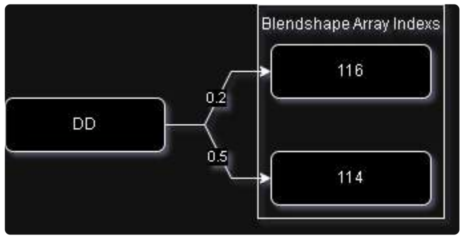
Lip Sync System

Convai sends Visemes or Blend Shape Frame from back-end depending upon the face model developer chooses to use and when return Convai SDK out of the box extracts and parses it and provides it to the `Convai LipSync Component`, after which the component relies of the `SkinMeshRen derer`'s `Blendshape Effectors` and `Bone Effectors` to give Convai powered NPC's realistic lipsync.

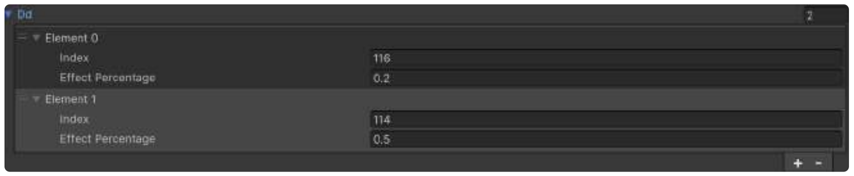
Components of LipSync System

Viseme Effector List

This is where developer will tell the Convai SDK, which index of Blendshape Array will be effector how much from which value. To better explain its working lets understand it with a diagram.



Here, its saying that whatever value is coming from the server will affect Blendshape at 116th index by 0.2 multiplier and Blendshape at 114th index by 0.5 multiplier. The engine representation of this would look something like this.

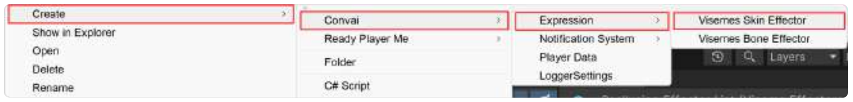


▼ Dd	2
▼ Element 0	
Index	116
Effect Percentage	0.2
▼ Element 1	
Index	114
Effect Percentage	0.5

So, you can make you own Effector list or use one of the many that we ship in the SDK.

How to Create your own Viseme Effector List

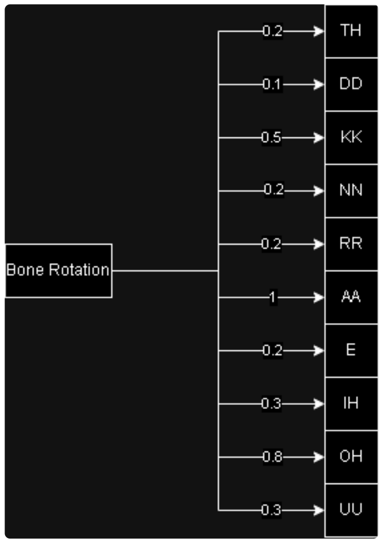
Right click inside project panel and head over to `Create > Convai > Expression > Viseme Skin Effector` which will create a **Viseme Effector List Scriptable Object** and now you can define your own values



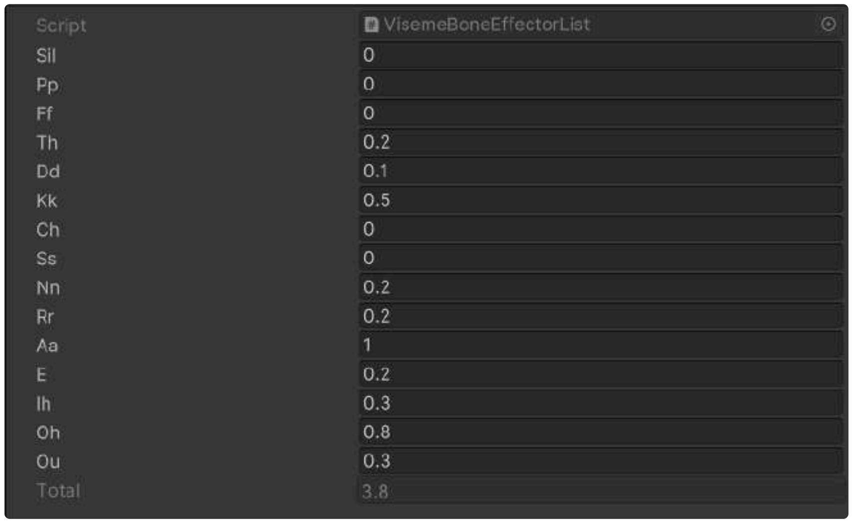
Viseme Bone Effector List

This is where developer will tell the Convai SDK, how much each value coming from server will affect the rotation of the bone. To better explain its working lets understand it with a diagram.

$$\begin{tikzcd}[row sep=small]
 & TH \\
 & DD \\
 & KK \\
 & NN \\
 & RR \\
\text{Bone Rotation} \arrow[ruuuuu, "0.2"] \arrow[ruuuu, "0.1"] \arrow[ruuu, "0.5"] \arrow[ruu, "0.2"] \arrow[ru, "0.2"] \arrow[r, "1"] \arrow[rd, "0.2"] \arrow[rdd, "0.3"] \arrow[rddd, "0.8"] \arrow[rdddd, "0.3"] & AA \\
 & E \\
 & IH \\
 & OH \\
 & UU
\end{tikzcd}$$



Here, bone's rotation will be effect by the values coming from server multiplied by the values in effects. Example, for TH the value will affect bone's rotation by 0.2 multiplier and etc. The engine representation of this would look something like this.



Script	VisemeBoneEffectorList
Sil	0
Pp	0
Ff	0
Th	0.2
Dd	0.1
Kk	0.5
Ch	0
Ss	0
Nn	0.2
Rr	0.2
Aa	1
E	0.2
Ih	0.3
Oh	0.8
Ou	0.3
Total	3.8

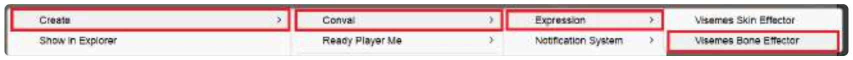
So, you can make you own Bone Effector list or use one of the many that we ship in the SDK.

We use this formula to calucate the roatation

```
UpdateJawBoneRotation(
new Vector3(
        0.0f,
        0.0f,
        -90.0f - CalculateBoneEffect(FacialExpressionData.JawBoneEffector
    )
);
UpdateTongueBoneRotation(
new Vector3(
        0.0f,
        0.0f,
        CalculateBoneEffect(FacialExpressionData.TongueBoneEffector) * 80
    )
);
```

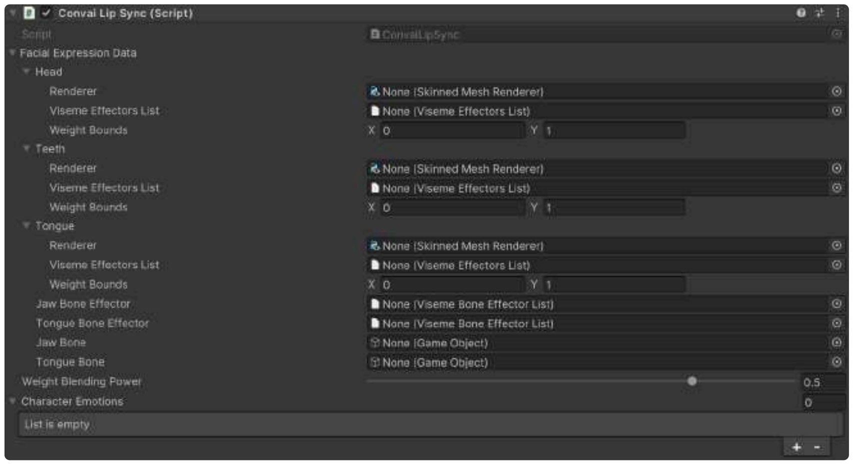
How to Create your own Viseme Bone Effector List

Right click inside project panel and head over to `Create > Convai > Expression > Viseme Bone Effector` which will create a **Viseme Bone Effector List Scriptable Object** and now you can define your own values.



Convai Lipsync Component

When you attach this component to your Convai Character, you will see something like this.



Convai Lip Sync (Script)

Script	ConvaiLipSync
Facial Expression Data	
Head	
Renderer	None (Skinned Mesh Renderer)
Viseme Effectors List	None (Viseme Effectors List)
Weight Bounds	X 0 Y 1
Teeth	
Renderer	None (Skinned Mesh Renderer)
Viseme Effectors List	None (Viseme Effectors List)
Weight Bounds	X 0 Y 1
Tongue	
Renderer	None (Skinned Mesh Renderer)
Viseme Effectors List	None (Viseme Effectors List)
Weight Bounds	X 0 Y 1
Jaw Bone Effector	None (Viseme Bone Effector List)
Tongue Bone Effector	None (Viseme Bone Effector List)
Jaw Bone	None (Game Object)
Tongue Bone	None (Game Object)
Weight Blending Power	0.5
Character Emotions	0
List is empty	

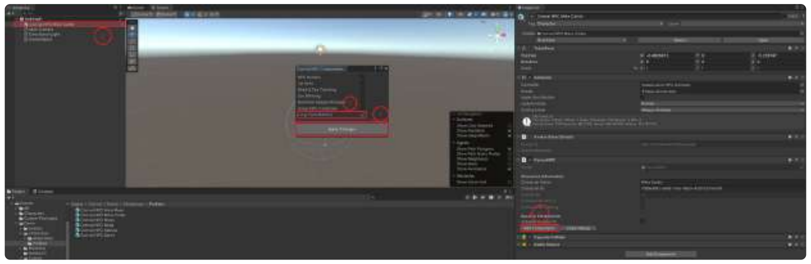
Let's learn what these learns are

1. Facial Expression Data
    1. Head | Teeth | Tongue
        1. Renderer: Skin Mesh Renderer which correspond to that specified part of the body
        2. Viseme Effectors List: How the SkinMeshRenderer's Blendshape will be affected by values coming from server.
    2. Jaw | Tongue Bone Effector
        1. How much Bone's rotation will be affected by values coming from server.
    3. Jaw | Tongue Bone
        1. Reference to the bone which control jaw and tongue respectively
2. Weight Blending Power
    1. Percentage to interpolate between two frames in late update.
3. Character Emotions
    1. Learn More about Character Emotions here

Steps to add Lipsync to your Convai Character

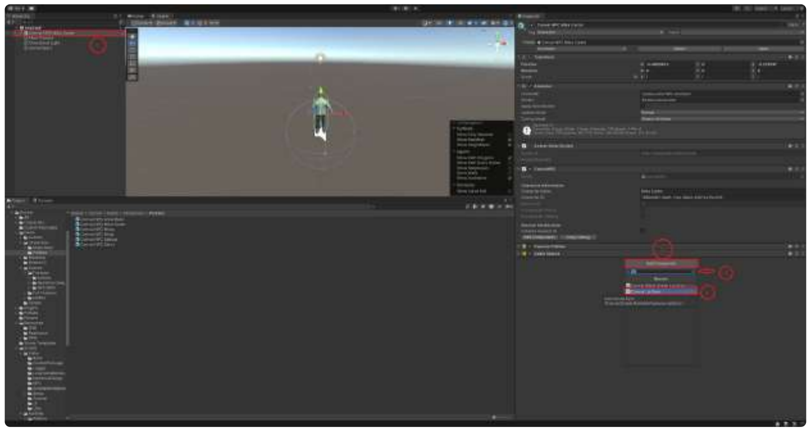
Using Convai Add Component Button

1. Select you Convai Powered Character in the hierarchy.
2. In the inspector panel search for ConvaiNPC component, there you will see Add Component Button.
3. Click on it and select Convai Lipsync Component and click on apply

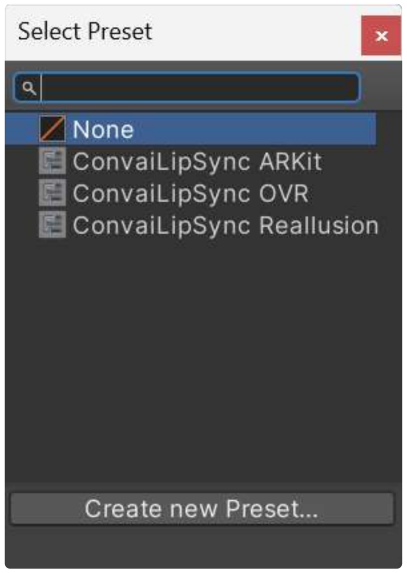


Using Unity Add Component Button

1. Select you Convai Powered Character in the hierarchy.
2. Click on Add Component
3. Search for Convai Lipsync
4. Select Convai Lipsync component



Now you can configure the Component according to your custom configuration or use one of the many Presets Convai ships with the SDK



Now your lipsync component would be ready to use in your application.

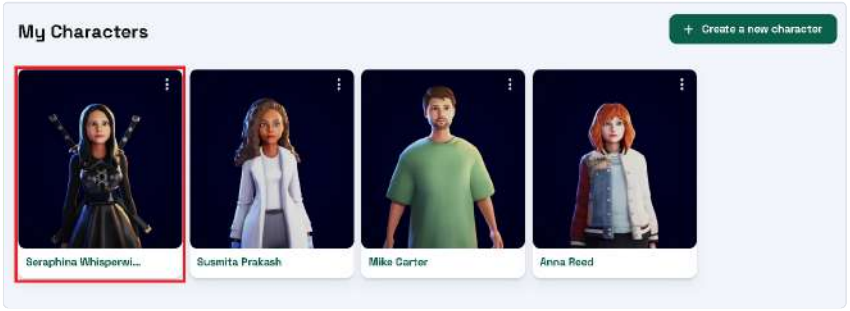
Adding Narrative Design to your Character

Follow this guide to incorporate Narrative Design into your Convai-powered characters. Follow this step-by-step tutorial, open your project, and let's begin!

Convai Playground

Step 1: Select your Character in which you want to enable Narrative Design

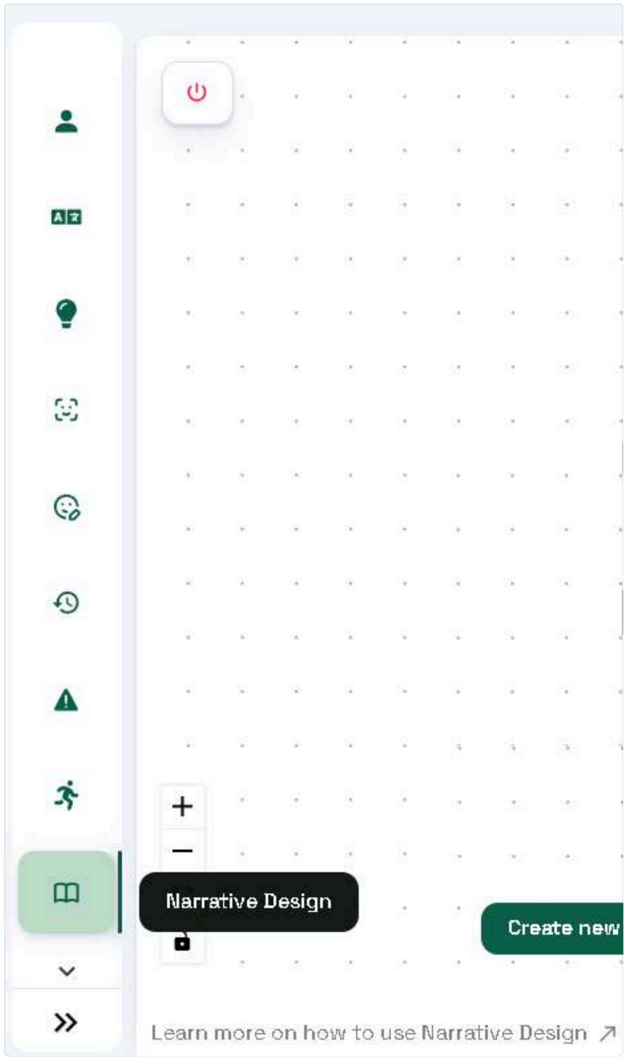
ⓘ For this demo, we are using `Seraphine Whisperwind`, you can select whatever character you want to enable Narrative Design.



Screenshot showing selection of character in Convai Playground

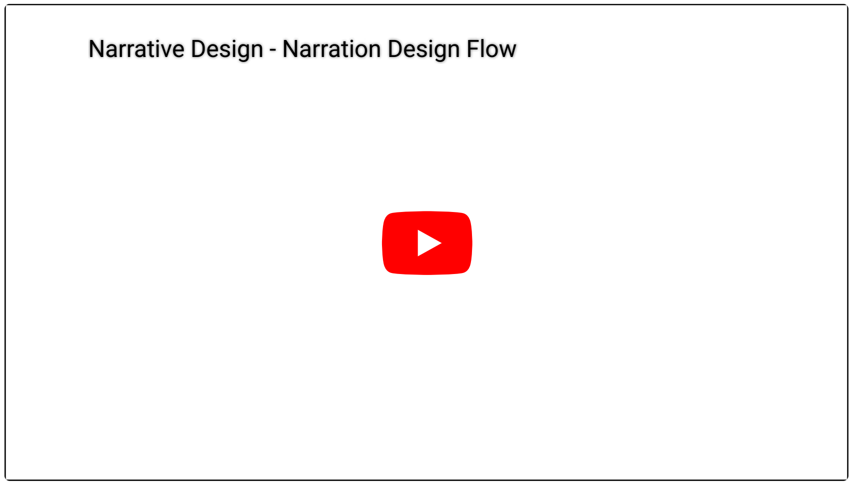
Step 2: Open Narrative Design in Convai Playground

Select the Narrative Design option from the side panel and create your narrative design



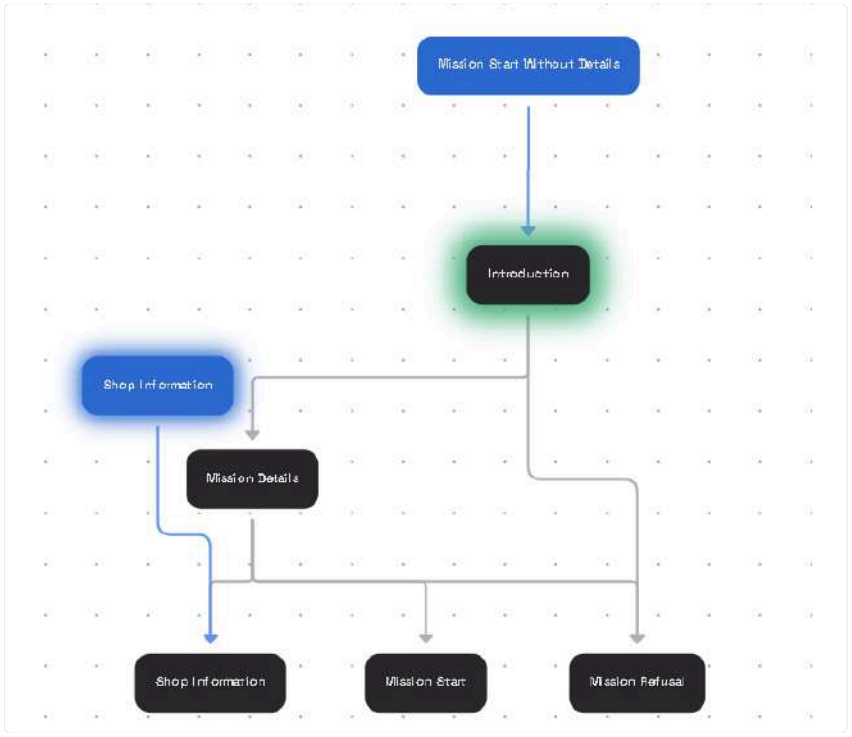
Screenshot showing Icon of Narrative Design

ⓘ For more information how to create narrative design in the Convai Playground please refer to the following YouTube video series



Video series showing how to create Narrative Design

For this sample we have created the following Narrative design



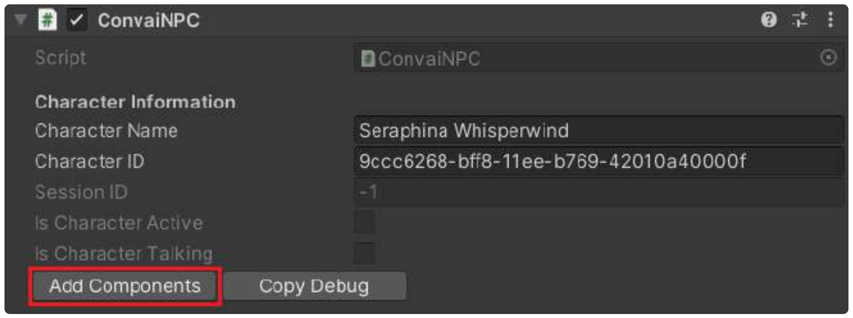
You are all set to bring your character from Convai Playground to Unity, let's hope over to Unity to continue the guide

Unity Setup

Step 1: Add the Narrative Design Manager Component

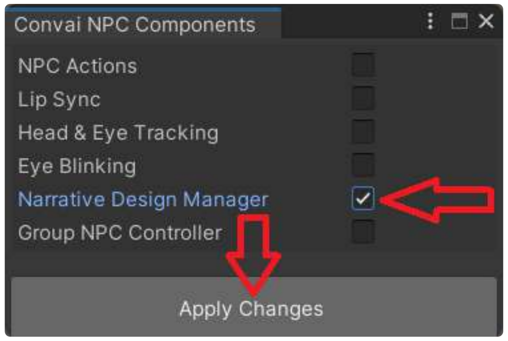
Using Add Components Button in Convai NPC (Recommended Way)

1: Select your Convai Character in the scene and look for ConvaiNPC component in the inspector panel. Click on Add Components button



Screenshot showing location of Add Components button in the Convai NPC inspector panel

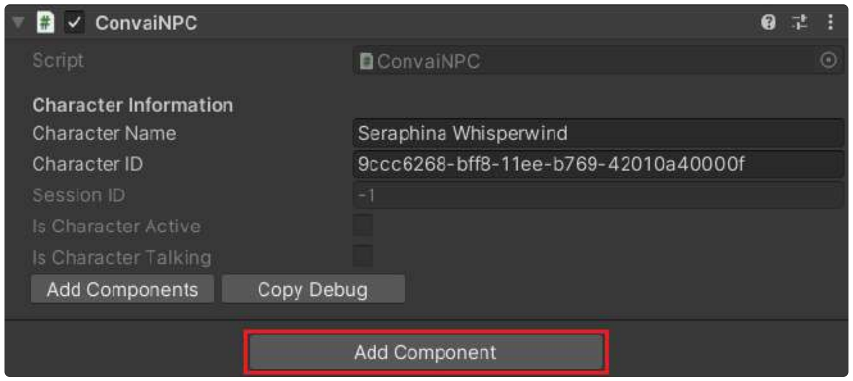
2: Select Narrative Design Manager checkbox and then click on Apply Changes button



Screenshot showing selection of Narrative design option in the Add Component Window

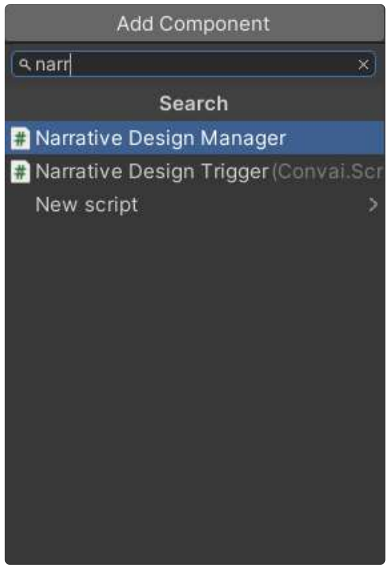
Using Unity Inspector

1: Select your Convai Character and find Add Component button in the inspector panel



Screenshot showing location of Add Component button in the inspector panel

2: Search for `Narrative Design Manager` in the search box and select it



Screenshot showing which component to select from the search results

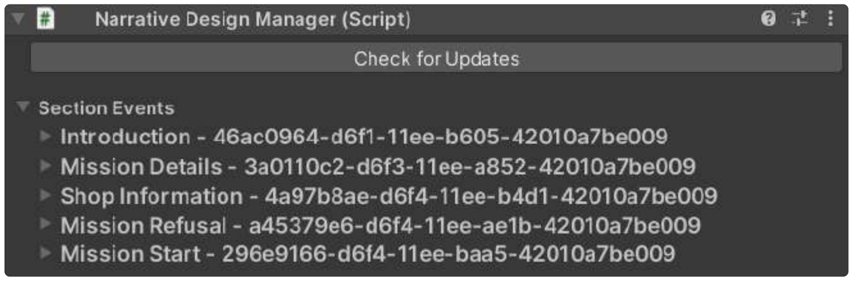
Step 2: Setup the Narrative Design Component

After adding the Narrative Design Component, you will be able to be the following component

This component system assumes that API key is setup correctly, so ensure that API key is setup correctly otherwise an error will be thrown.

After adding, component will retrieve the sections for the character ID taken from the ConvaiNPC, please wait for some time depending upon your network speed

The following section events are for character used in demo, and you will see section events corresponding to your character in which Narrative Design is enabled.


Narrative Design Manager (Script)

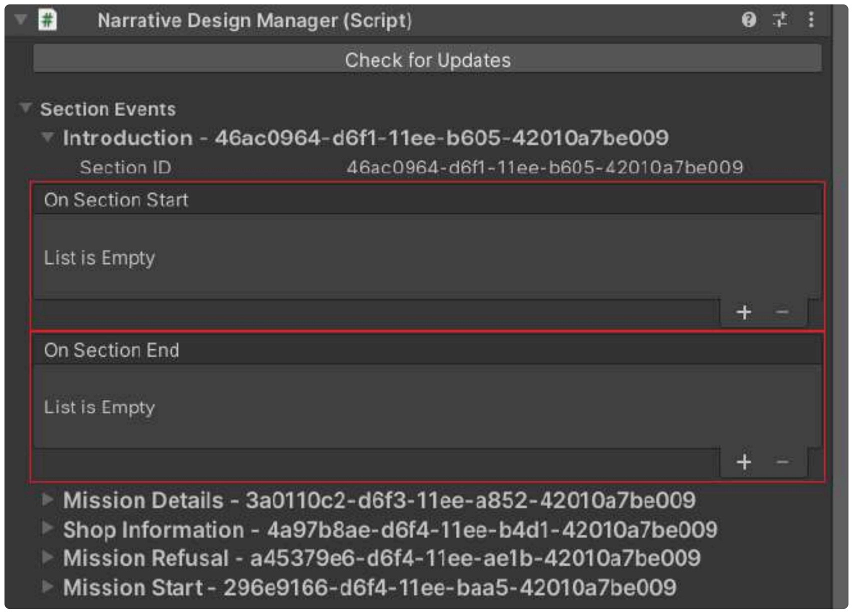
Check for Updates

Section Events
- Introduction - 46ac0964-d6f1-11ee-b605-42010a7be009
- Mission Details - 3a0110c2-d6f3-11ee-a852-42010a7be009
- Shop Information - 4a97b8ae-d6f4-11ee-b4d1-42010a7be009
- Mission Refusal - a45379e6-d6f4-11ee-ae1b-42010a7be009
- Mission Start - 296e9166-d6f4-11ee-baa5-42010a7be009

Screenshot showing a sample Narrative Design component

Getting to know the Narrative Design Component

Expanding the section event, you will see two unity events you can subscribe to, one is triggered when section starts, and another one is triggered when section ends


Narrative Design Manager (Script)

Check for Updates

Section Events

Introduction - 46ac0964-d6f1-11ee-b605-42010a7be009

Section ID 46ac0964-d6f1-11ee-b605-42010a7be009

On Section Start

List is Empty

On Section End

List is Empty

- Mission Details - 3a0110c2-d6f3-11ee-a852-42010a7be009
- Shop Information - 4a97b8ae-d6f4-11ee-b4d1-42010a7be009
- Mission Refusal - a45379e6-d6f4-11ee-ae1b-42010a7be009
- Mission Start - 296e9166-d6f4-11ee-baa5-42010a7be009

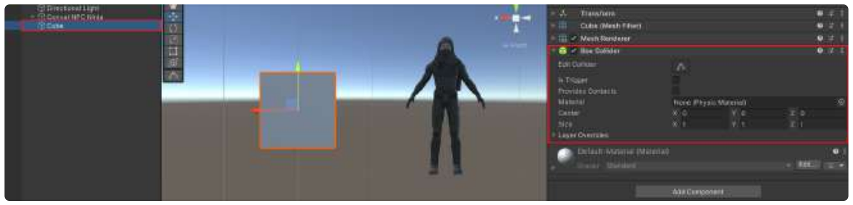
Screenshot showing various unity events user can subscribe to

Getting to know about Section Triggers

Section triggers are a way to directly invoke a section in narrative design and can be used to jump to a different section in your narrative design

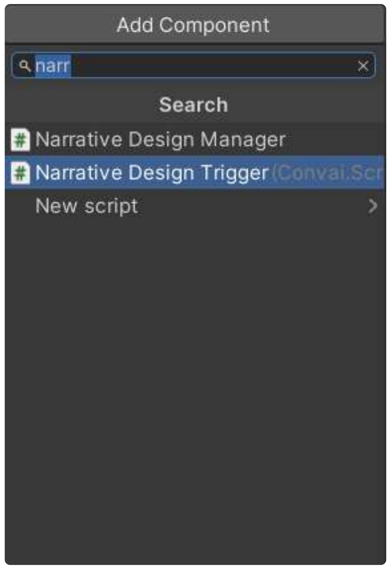
Step 1: Select the game object you want to make a trigger, in this example we have selected a simple cube, but it's up to your imagination.

Make sure that game object you have decided to be a trigger have a collider attach to it



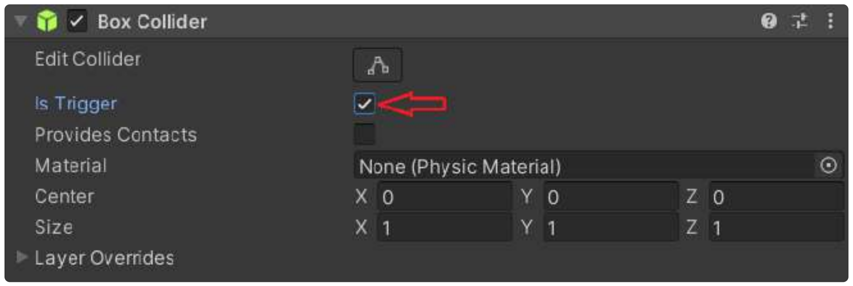
Screenshot showing a game object with a collider selected

Step 2: Add Narrative design Trigger from Add Component menu by searching for it



Screenshot showing selection of Narrative Design Trigger

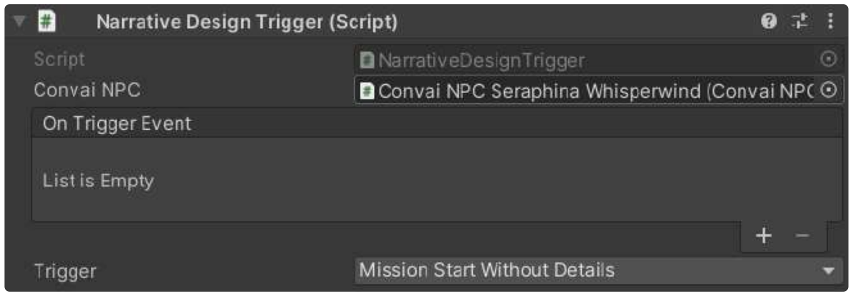
Step 3: Make the collider a trigger.



Box Collider
Edit Collider
Is Trigger ✓
Provides Contacts
Material None (Physic Material)
Center X 0 Y 0 Z 0
Size X 1 Y 1 Z 1
Layer Overrides

Screenshot showing Box Collider becoming a trigger box

Step 4: Assign your Convai NPC to Convai NPC field

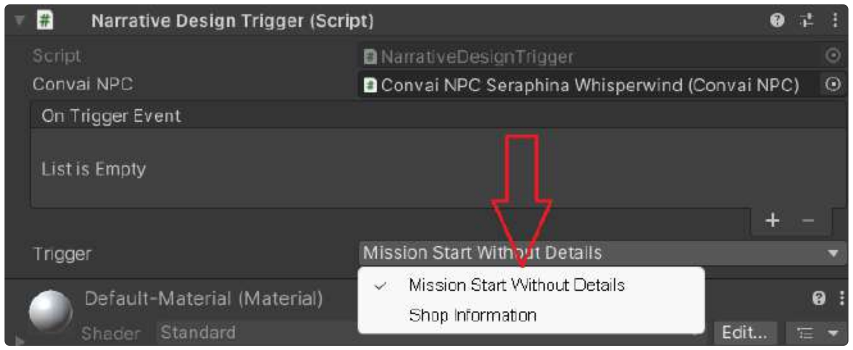


Narrative Design Trigger (Script)
Script NarrativeDesignTrigger
Convai NPC Convai NPC Seraphina Whisperwind (Convai NPC
On Trigger Event
List is Empty
Trigger Mission Start Without Details

Screenshot showing assigning of Convai NPC to trigger component

Now you can select from the "Trigger" dropdown which trigger should be invoked when player enters this trigger box.

We have added a way for you to manually invoke this trigger also, you can use `InvokeSelectedTrigger` function to invoke the trigger from any where


Narrative Design Trigger (Script)
Script: NarrativeDesignTrigger
Convai NPC: Convai NPC Seraphina Whisperwind (Convai NPC)
On Trigger Event
List is Empty
Trigger: Mission Start Without Details

Screenshot showing ability to select your desired trigger

Invoke Trigger from any script

You can use this code block as a reference to invoke the trigger from anywhere

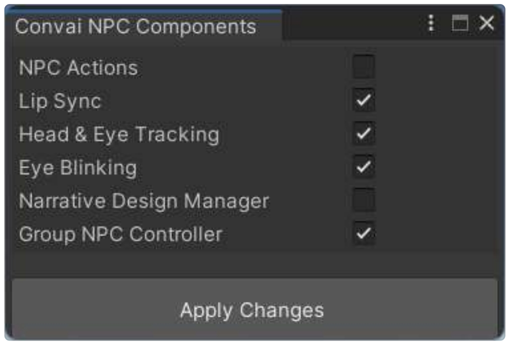
```
if(convaiNPC.TryGetComponent(out NarrativeDesignTrigger narrativeDesignTr
{
    //Optional message parameter if you want to send some message while i
    //the trigger
    string message = "Player has collected enough resources";
    narrativeDesignTrigger.InvokeSelectedTrigger(message);
}
```

Adding NPC to NPC Conversation

This guide will walk you through setting up the NPC to NPC conversation feature in the Convai SDK.

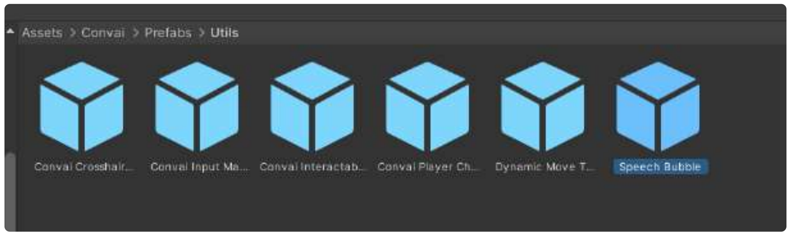
Step 1: Setting up Convai NPC

1. **Go to your Convai NPCs**:
   - Select the NPCs you want to include in the conversation.
2. **Enable Group NPC Controller**:
   - Click on the `Group NPC Controller` checkbox in the inspector panel.
   - Click `Apply Changes` to add the group NPC controller script.


Convai NPC Components

NPC Actions	
Lip Sync	✓
Head & Eye Tracking	✓
Eye Blinking	✓
Narrative Design Manager	
Group NPC Controller	✓

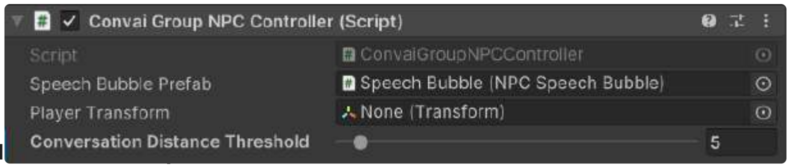
3. **Create or Find the Speech Bubble Prefab**:
   - Create a new speech bubble prefab or use the one provided in the `Prefabs` folder.



4. **Attach Required Components:**
   -

Add the speech bubble prefab and the player transform (optional, defaults to the main camera if not provided).

- Set the conversation distance threshold variable (set it to zero to disable this feature, meaning NPC to NPC conversations will always happen regardless of the player's distance).



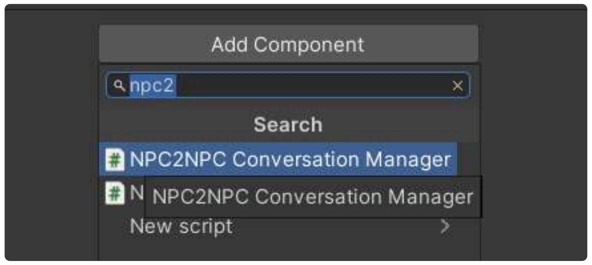
Convai Group NPC Controller (Script)

Script	ConvaiGroupNPCController
Speech Bubble Prefab	Speech Bubble (NPC Speech Bubble)
Player Transform	None (Transform)
Conversation Distance Threshold	5

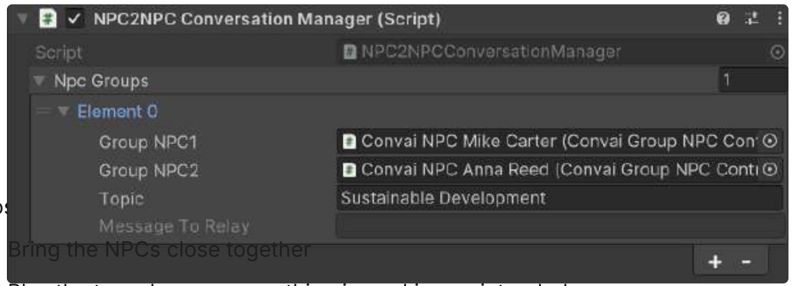
5. **Ad**
   - Add components like lip sync, eye and head tracking, character blinking, etc., to the Convai NPC.

Step 2: Setting up NPC Manager

1. **Create an NPC To NPC Manager GameObject**:
   - Add an empty GameObject and rename it to `NPC to NPC Manager` (optional).
2. **Add the NPC2NPC Conversation Manager Script**:
   - Attach the `NPC2NPCConversationManager` script to the GameObject.



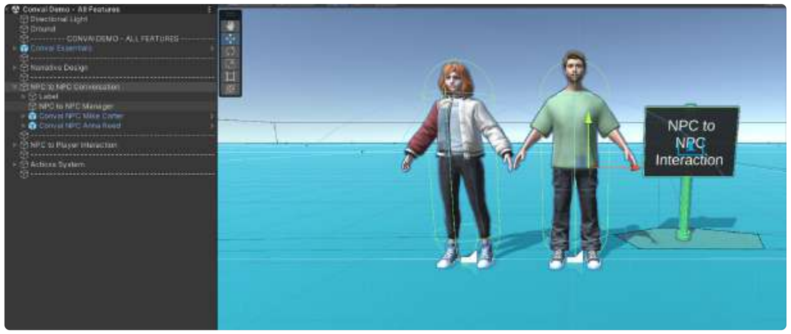
3. **Configure the NPC Group List**:
   - In the `NPC Group List`, click on the `+` icon to add a new list element.
   - Add the NPCs you want to include in the group conversation.
   - Set the group discussion topic.



NPC2NPC Conversation Manager (Script)

Script	NPC2NPCConversationManager
Npc Groups	1
Element 0	
Group NPC1	Convai NPC Mike Carter (Convai Group NPC Con
Group NPC2	Convai NPC Anna Reed (Convai Group NPC Conti
Topic	Sustainable Development
Message To Relay	

4. **Pos**
   - Bring the NPCs close together
   - Play the to make sure everything is working as intended.

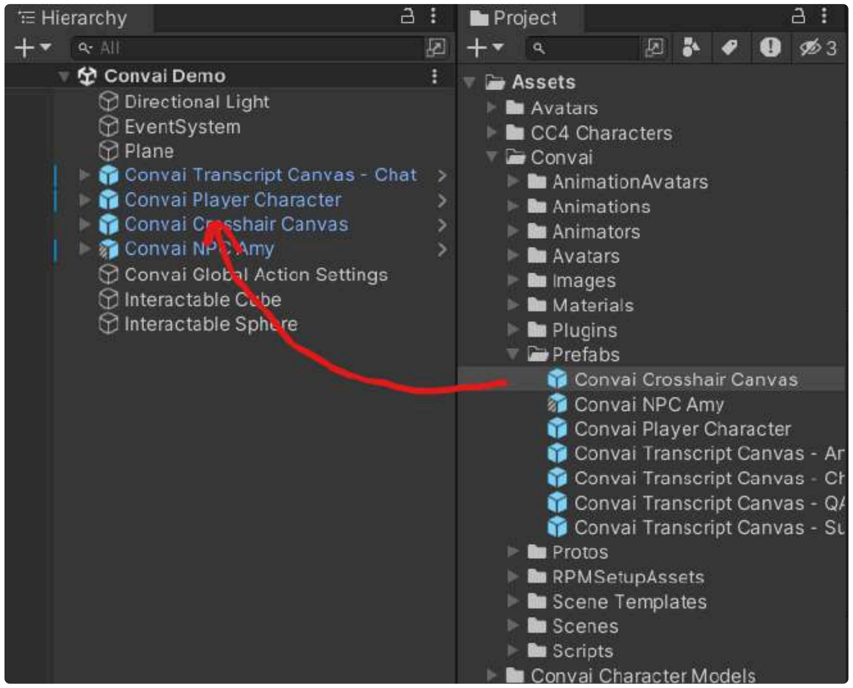


By following these steps you can set up and manage NPC to NPC conversations in your Convai-powered application. For further customization and integration, refer to the complete implementation code and adjust it as needed for your specific use case.

Adding Scene Reference and Point-At Crosshairs

You can point at Interactable Objects and Characters and ask your characters about them.

To enable this, simply drag and drop the `Convai Crosshair Canvas` prefab into the scene.



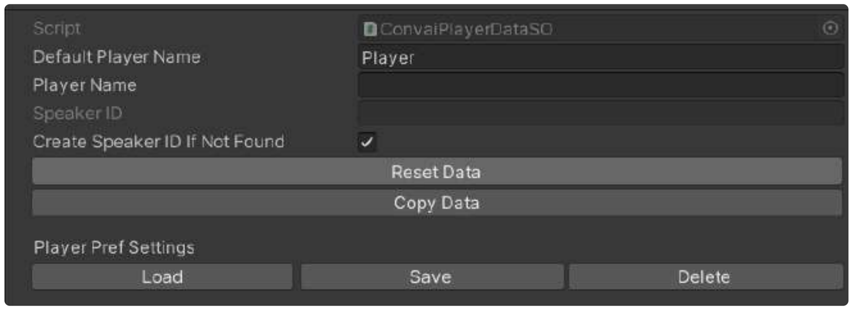
Utilities

Unity Plugin Utilities - Enhance development with Convai's tools and resources.

Player Data Container

All the information that Convai SDK needs from the player to work properly

This is a scriptable object which is made automatically after you hit play in the editor with Convai SDK installed and in a Scene where Convai Base Scene Essentials Prefab is present


Script: ConvaiPlayerDataSO
Default Player Name: Player
Player Name:
Speaker ID:
Create Speaker ID If Not Found: ✓
Reset Data
Copy Data
Player Pref Settings
Load | Save | Delete

Default Player Name

You can provide a default name of your players.

Player Name

Current name of your player, out of the box if you use our settings panel, we keep it updated automatically, if you are using some custom logic, it will be your responsibility to keep it updated, as our transcript UI use this name to show it in UI

Speaker ID

Speaker ID for the player. **Please note that Speaker ID is directly linked with your API key, so for each API key there should be a unique speaker ID associated with it.** We handle the creation of the Speaker ID when it's not found in the Player Prefs if the Boolean is set to true.

Create Speaker ID If Not Found

This Boolean lets the SDK if it should create a unique Speaker ID for that Player Name if it is not found in the Player Prefs.

Buttons

Reset Data

It just makes the **Player Name** and **Speaker ID** fields empty.

Copy Data

Copies the data into system buffer so you can paste it anywhere for debugging purpose

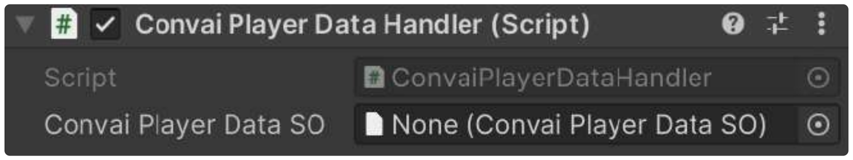
Player Pref Settings Button

1. **Load:** Loads the **Player Name** and associated **Speaker ID** from the player Prefs
2. **Save:** Saves the **Player Name** and associated **Speaker ID** from the player Prefs
3. **Delete:** Deletes the **Player Name** and associated **Speaker ID** from the player Prefs

How to maintain the Player Data

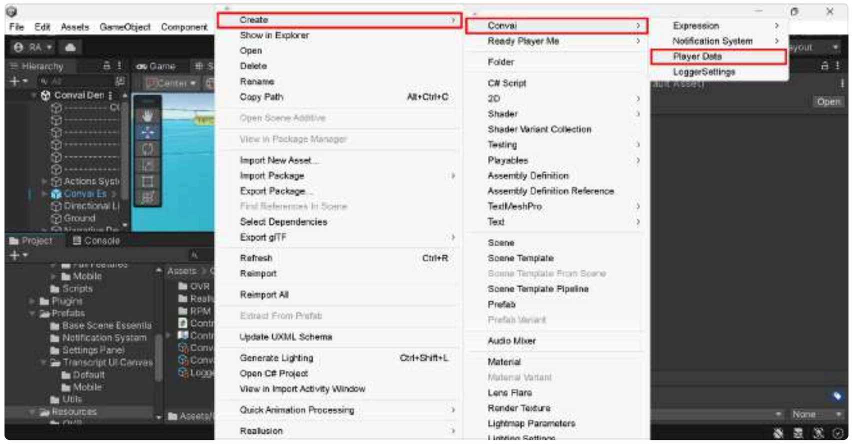
Convai provides a pre-made component which you can add to any `GameObject` to make the `PlayerDataContainer` work out of the box.

Choose an existing GameObject or create a new `GameObject` in the scene and add the `ConvaiPlayerDataHandler` component to your chosen `GameObject` and it should start working


Convai Player Data Handler (Script)
Script: ConvaiPlayerDataHandler
Convai Player Data SO: None (Convai Player Data SO)

Optional Step

You can also create the required Scriptable Object by going to `Assets > Convai > Resources` and right clicking in the project panel and navigating to `Create > Convai > Player Data` and name it `ConvaiPlayerDataSO`



⚠ Make sure you name the created Scriptable Object exactly `ConvaiPlayerDataSO` as our system looks for this exact name

Long Term Memory

Learn how to enable character retain conversation history across multiple sessions

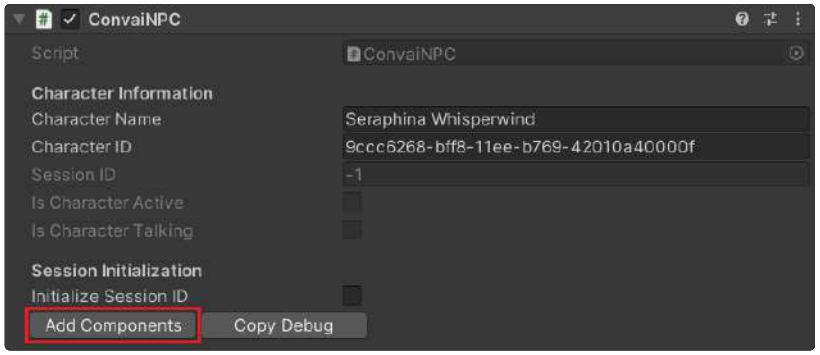
Long-Term Memory (LTM) enables the persistent storage of conversational history with NPCs, allowing players to seamlessly continue interactions from where they previously left off, even across multiple sessions. This feature significantly enhances the realism of NPCs, aligning with our goal of creating more immersive and lifelike characters within your game.

ⓘ Prerequisite: Have a project with Convai SDK version 3.1.0 or higher. If you don't have it, check this documentation

[Setting Up Unity Plugin]

Steps to get LTM working

1. Select your Convai Character



ConvaiNPC

Script: ConvaiNPC

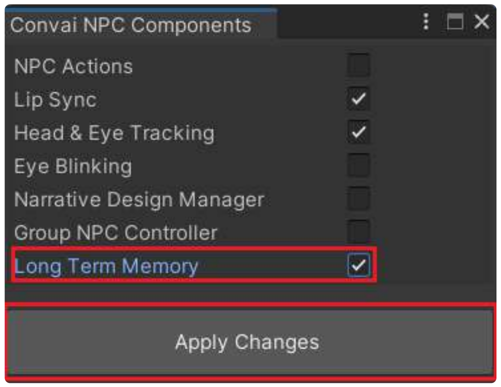
Character Information

Character Name: Seraphina Whisperwind
Character ID: 9ccc6268-bff8-11ee-b769-42010a40000f
Session ID: -1
Is Character Active
Is Character Talking

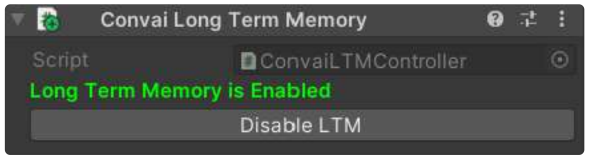
Session Initialization

Initialize Session ID

2. Add the Long-Term Memory Component onto your character



3. Make sure that Long Term Memory is enabled for that character

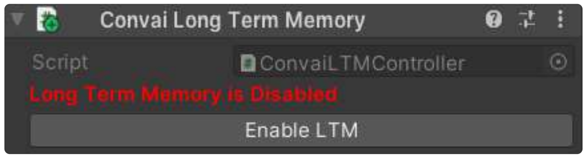


Long Term Memory should now be working for your character.

Components of the LTM System

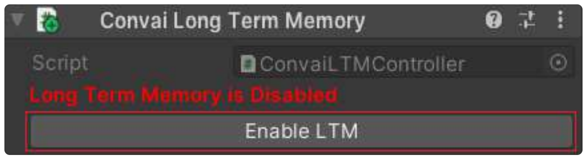
Convai Long Term Memory Component

This component will enable or disable LTM right from the unity editor

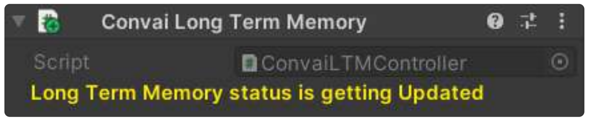


Toggling Long Term Memory

1) Click the button provided in the component



2) It will take some time to update, and after that the new status of the LTM should be visible in the inspector.



Since enabling or disabling Long-Term Memory (LTM) for a character is a global action that impacts all players interacting with that character, we strongly recommend against toggling the LTM status at runtime. This functionality should be managed exclusively by developers or designers through the editor to ensure consistent gameplay experiences.

Troubleshooting

Grpc.Core.RpcException: Status(StatusCode=InvalidArgument, Detail="Cannot find speaker with id: 99fbef96-5ecb-11ef-93ce-42010a7be011.")

If you encounter this error, ensure that the **SpeakerID** was created using the same API key currently in use. If you're uncertain about the API key used, you can reset the **SpeakerID** and **PlayerName** by accessing the `ConvaiPlayerDataSO` file located in `Assets > Convai > Resources`, allowing you to start the process anew.

Managing sessionID Locally

Session ID Management - Manage unique session IDs for Convai Unity integration.

In a typical application integrating with the Convai API, maintaining a consistent session ID across different sessions is crucial for providing a seamless user experience. This documentation outlines the best practices for storing and retrieving session IDs using Unity's `PlayerPrefs`, including detailed steps and example scripts.

Importance of Session IDs

A session ID uniquely identifies a session between the client and the Convai server. Storing the session ID locally ensures that the same session ID is used across different sessions, which helps in maintaining context and continuity in interactions.

Storing Session IDs

When initializing a session, if a session ID is not available locally, it should be fetched from the server and then stored locally for future use. Here's how you can achieve this:

1. **Fetch and Store Session ID**: When initializing a session, check if a session ID is stored locally. If not, fetch a new session ID from the server and store it using `PlayerPrefs`.

```
public static async Task<string> InitializeSessionIDAsync(string characte
{
    // Retrieve stored session ID if it exists
    string sessionID = PlayerPrefs.GetString(characterID, string.Empty);

    // If no session ID is stored, initialize a new one
    if (string.IsNullOrEmpty(sessionID))
    {
        sessionID = await ConvaiGRPCAPI.InitializeSessionIDAsync(characte

        // Store the new session ID locally
        if (!string.IsNullOrEmpty(sessionID))
        {
            PlayerPrefs.SetString(characterID, sessionID);
            PlayerPrefs.Save();
        }
    }

    return sessionID;
}
```

Retrieving Session IDs

When initializing your application, retrieve the stored session ID to ensure continuity in user interactions.

```
private async void Start()
{
    // Initialize session ID on start
    string characterID = "YourCharacterID"; // Replace with your actual cl
    string sessionID = await InitializeSessionIDAsync("CharacterName", gr

    if (!string.IsNullOrEmpty(sessionID))
    {
        Debug.Log("Session ID initialized and stored: " + sessionID);
    }
    else
    {
        Debug.LogError("Failed to initialize session ID.");
    }
}
```

Example Class for Session Management

The following example class demonstrates how to manage session IDs using `PlayerPrefs` in a Unity project:

```
using System;
using System.Threading.Tasks;
using Convai.Scripts.Utils;
using Google.Protobuf;
using Grpc.Core;
using Service;
using UnityEngine;
using static Service.GetResponseRequest.Types;

public class SessionManager : MonoBehaviour
{
    public ConvaiService.ConvaiServiceClient grpcClient;

    private void Start()
    {
        // Initialize session ID on start
        InitializeSession("CharacterName", grpcClient, "YourCharacterID")
    }

    private async void InitializeSession(string characterName, ConvaiServ
    {
        string sessionID = await InitializeSessionIDAsync(characterName,

        if (!string.IsNullOrEmpty(sessionID))
        {
            Debug.Log("Session ID initialized and stored: " + sessionID);
        }
        else
        {
            Debug.LogError("Failed to initialize session ID.");
        }
    }

    public static async Task<string> InitializeSessionIDAsync(string chara
    {
        string sessionID = PlayerPrefs.GetString(characterID, string.Empt

        if (string.IsNullOrEmpty(sessionID))
        {
            sessionID = await ConvaiGRPCAPI.InitializeSessionIDAsync(chara

            if (!string.IsNullOrEmpty(sessionID))
            {
                PlayerPrefs.SetString(characterID, sessionID);
                PlayerPrefs.Save();
            }
        }
        return sessionID;
    }
}
```

Detailed Steps for Session Management

1. **Initialize Session**: Call `InitializeSessionIDAsync` to check if a session ID is stored. If not, fetch and store it.
2. **Store Session ID**: Use `PlayerPrefs.SetString(characterID, sessionID)` to store the session ID locally.
3. **Retrieve Session ID**: Use `PlayerPrefs.GetString(characterID, string.Empty)` to retrieve the stored session ID.
4. **Use Session ID**: Pass the session ID to your Convai API calls to maintain session continuity.

Best Practices

- **Error Handling**: Ensure proper error handling when fetching and storing session IDs.
- **Security**: Consider encrypting sensitive information stored in `PlayerPrefs`.
- **Performance**: Use asynchronous methods to avoid blocking the main thread when fetching session IDs.

Transcript UI System

Transcript UI System - Integrate transcript UI with Convai's Unity plugin.

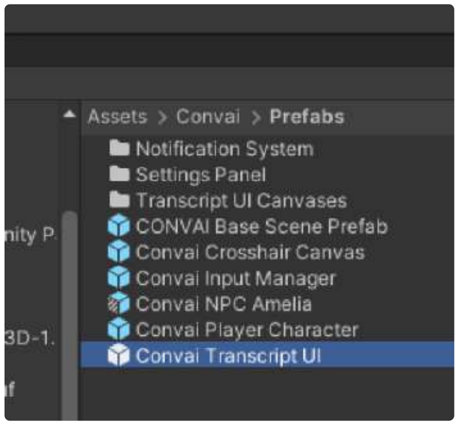
Overview

The Dynamic UI system is a feature within the Convai Unity SDK that provides developers a robust system for in-game communication. This feature allows for displaying messages from characters and players and supports various UI components for chat, Q&A sessions, subtitles, and custom UI types. This document will guide you through the integration, usage, and creation of custom UI types of the Dynamic UI feature in your Unity project.

Usage

Accessing the Chat UI Handler

To interact with the chat system, you need to reference the `ConvaiChatUIHandler` in your scripts. You can find the Transcript UI prefab in the Prefabs folder.



Here's an example of how to find and assign the handler:

```
private ConvaiChatUIHandler _convaiChatUIHandler;

private void OnEnable()
{
    // Find and assign the ConvaiChatUIHandler component in the scene
    _convaiChatUIHandler = ConvaiChatUIHandler.Instance;
    if (_convaiChatUIHandler != null) _convaiChatUIHandler.UpdateCharacte
}
```

Sending Messages

Once you have a reference to the `ConvaiChatUIHandler`, you can send messages using the following methods:

Sending Player Text

To send text as the player:

```
_convaiChatUIHandler.SendPlayerText(input);
```

- `input`: The string containing the player's message.

Sending Character Text

To send text as a character:

```
_convaiChatUIHandler.SendCharacterText(characterName,
currentResponseAudio.AudioTranscript.Trim());
```

- `characterName`: The name of the character sending the message.
- `currentResponseAudio.AudioTranscript`: The transcript of the audio response from the character, trimmed of any leading or trailing whitespace.

Adding Custom UI Types to the Dynamic Chatbox

While the Dynamic UI system within the SDK provides several pre-built UI types, you may want to create a custom UI that better fits the style and needs of your game and it designed to be extensible, allowing developers to add their custom UI types. This is achieved by inheriting from the `ChatUIBase` class and implementing the required methods. The `ConvaiChatUIHandler` manages the different UI types and provides a system to switch between them.

Creating a Custom UI Class

To create a custom UI type, follow these steps:

Step 1: Define Your Custom Class

Create a new C# script in your Unity project and define your class to inherit from `ChatUIBase`. For example:

```
using Convai.Scripts.Utils;
using UnityEngine;

public class CustomChatUI : ChatUIBase
{
    // Implement the required methods from ChatUIBase here.
}
```

Step 2: Implement Required Methods

Implement the abstract methods from `ChatUIBase`. You must provide implementations for `Initialize`, `SendCharacterText`, and `SendPlayerText`:

```
public override void Initialize(GameObject uiPrefab)
{
    // Instantiate and set up your custom UI prefab here.
}

public override void SendCharacterText(string charName, string
text, Color characterTextColor)
{
    // Handle sending character text to your custom UI here.
}

public override void SendPlayerText(string playerName, string text,
Color playerTextColor)
{
    Handle sending player text to your custom UI here.
}
```

Step 3: Add Custom Functionality

Add any additional functionality or customization options that your custom UI may require.

Step 4: Assign and Use Your Custom UI

To use your custom UI class within the dictionary `ConvaiChatUIHandler`, you need to add it to the `GetUIAppearances` dictionary. This involves creating a prefab for your custom UI and assigning it in the `ConvaiChatUIHandler`.

Here's an example of how to do this:

1. Create a prefab for your custom UI and add your `CustomChatUI` component to it.
2. Assign the prefab to a public variable in the `ConvaiChatUIHandler` script.
3. Modify the `InitializeUIStrategies` method in the `ConvaiChatUIHandler` script to include your custom UI type.

```
[Tooltip (Prefab for the customChatUI.")]
public GameObject customChatUIPrefab;

private void InitializeUIStrategies()
{
    Existing UI types
    InitializeUI(chatBoxPrefab, UIType.ChatBox);
    InitializeUI(questionAnswerPrefab, UIType.QuestionAnswer);
    InitializeUI(subtitlePrefab, UIType.Subtitle);

    // Custom UI type
    InitializeUI(customChatUIPrefab, UIType.Custom); // Make sure
to define UIType.Custom in the UIType enum
}

private void InitializeUI (GameObject uiPrefab, UIType uiType)
{
    // existing code...

    Add your custom UI initialization here
    if (uiType == UIType.Custom)
    {
        CustomChatUI customUIComponent =
uiPrefab.GetComponent<CustomChatUI>();
        if (customUIComponent == null)
        {
            Debug.LogError("CustomChatUI component not found on
prefab.");
            return;
        }

        customUIComponent.Initialize(uiPrefab);
        GetUIAppearances[uiType] = customUIComponent;
    }
}
```

4. Ensure that your custom UI type is added to the `UIType` enum:

```
public enum UIType
{
    ChatBox,
    QuestionAnswer,
    Subtitle,
    CustomUI // Your custom UI type
}
```

5. Now you can set your custom UI type as the active UI from the Settings Panel Settings Panel.

By following these steps, you can integrate your custom UI type into the Dynamic Chatbox system and switch between different UI types at runtime.

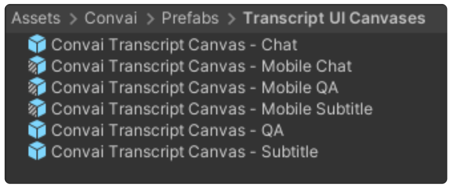
Pre-built UI Prefabs

Convai UI Prefabs - Utilize ready-to-use UI elements for Convai integration.

We provide several UI options to display character and user's transcript out of the box that players can use with the Convai Plugin. You can use and customize these prefabs.

The ConvaiNPC and ConvaiGRPCAPI scripts look for GameObjects with Convai Chat UI Handler as a component, and send any transcripts to the script so that it can be displayed on screen.

Types of UI



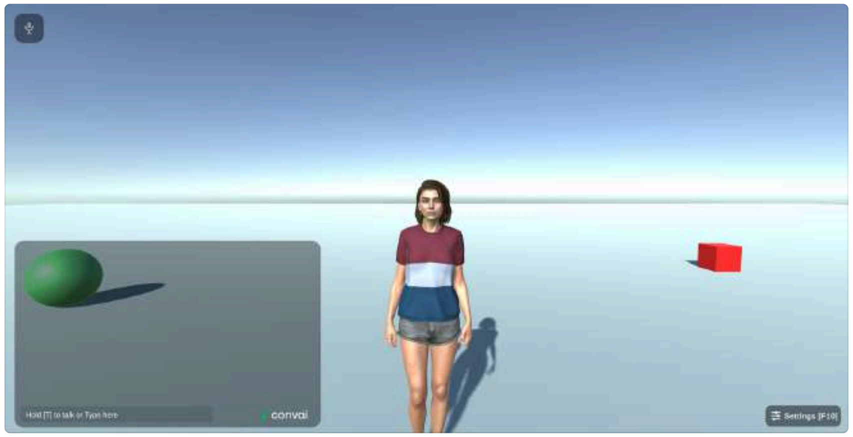
Assets > Convai > Prefabs > Transcript UI Canvases

- Convai Transcript Canvas - Chat
- Convai Transcript Canvas - Mobile Chat
- Convai Transcript Canvas - Mobile QA
- Convai Transcript Canvas - Mobile Subtitle
- Convai Transcript Canvas - QA
- Convai Transcript Canvas - Subtitle

ChatBox

Prefab Name: Convai Transcript Canvas - Chat

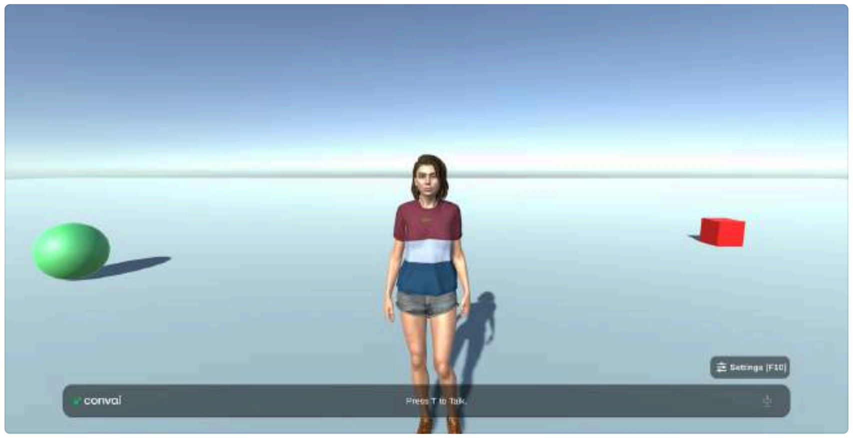
Both the user's and the character's transcripts are displayed one after other in a scrollable chat box.



Subtitle

Prefab Name: Convai Transcript Canvas - Subtitle

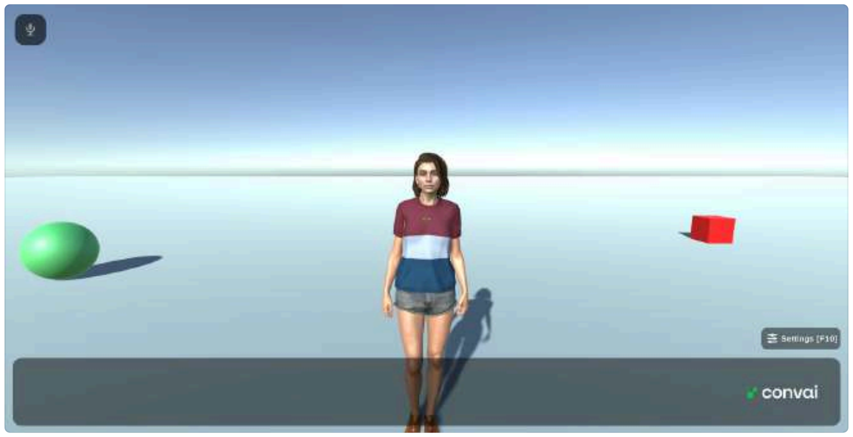
The user and character transcripts are displayed in the bottom like subtitles.



Question-Answer

Prefab Name: Convai Transcript Canvas - QA

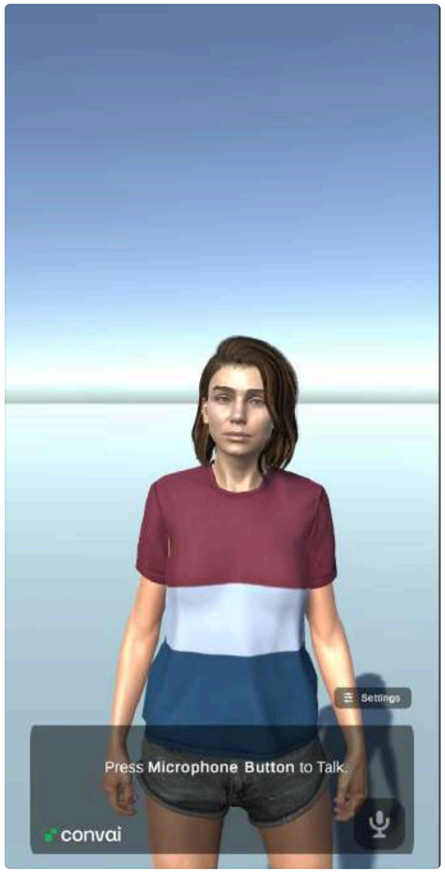
The user's transcript is displayed in the top where as the character's transcript is displayed in the bottom.



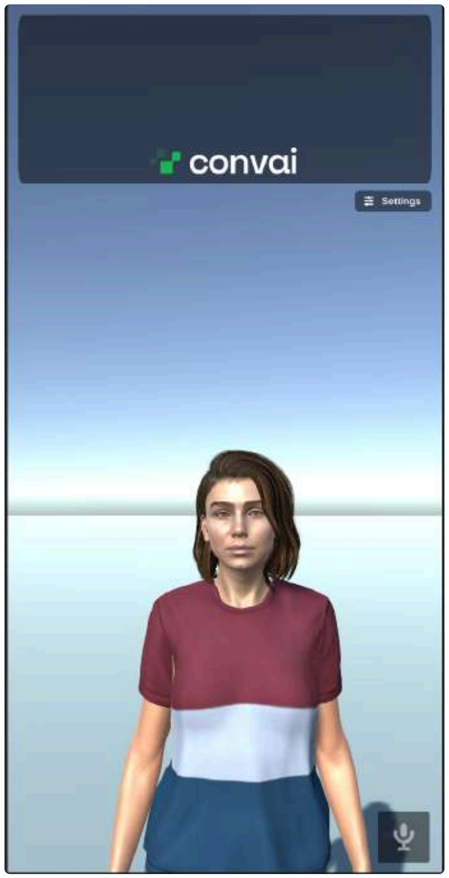
Mobile Optimised UI Styles

Prefab Name: Convai Transcript Canvas - Mobile Subtitle

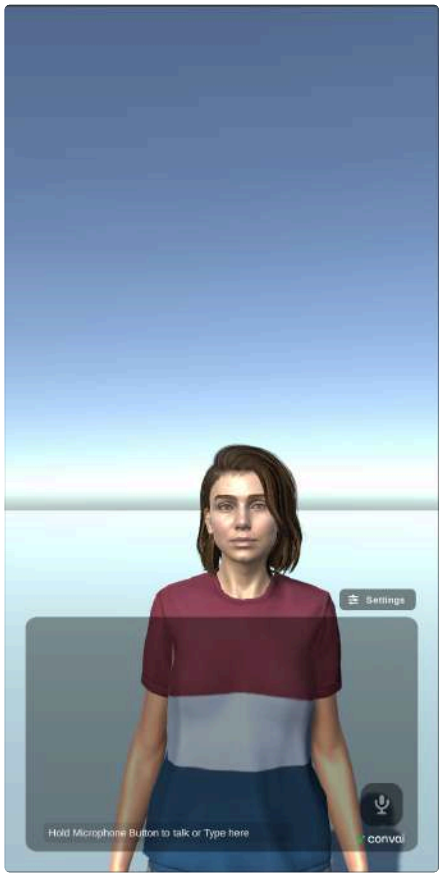
Identical to #subtitle UI. Includes a button that can be pressed and held for the user to speak. Ideal for portrait orientation of screen.



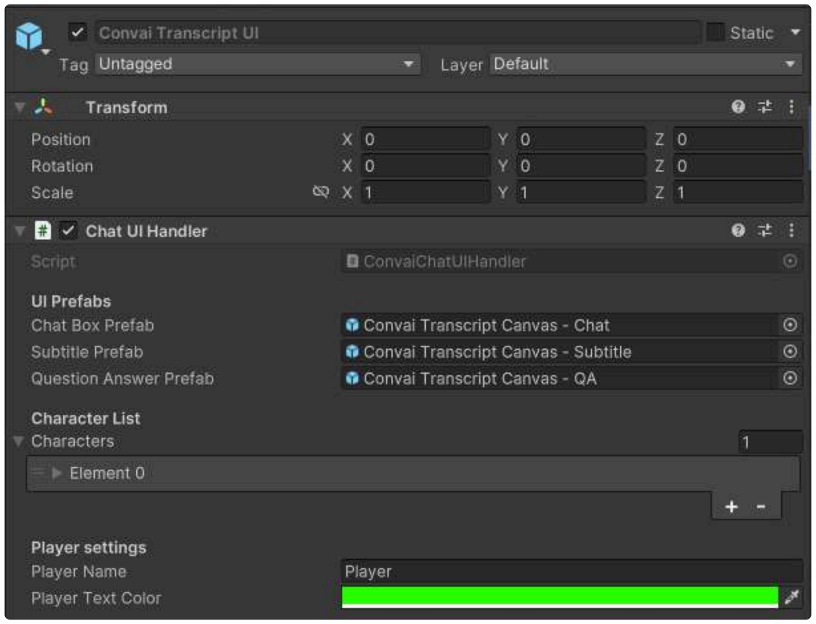
Prefab Name: Convai Transcript Canvas - Mobile QA



Prefab Name: Convai Transcript Canvas - Mobile Chat



Convai Chat UI Handler Component


Convai Transcript UI — Static
Tag: Untagged — Layer: Default

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1

Chat UI Handler
Script: ConvaiChatUIHandler

UI Prefabs
Chat Box Prefab: Convai Transcript Canvas - Chat
Subtitle Prefab: Convai Transcript Canvas - Subtitle
Question Answer Prefab: Convai Transcript Canvas - QA

Character List
Characters: 1
Element 0

Player settings
Player Name: Player
Player Text Color: 🟩

Functions to Know

SendCharacterText	A public function that sends a string of text to be displayed as character transcript along with the name of the character who said it.
SendUserText	A public function that sends a string of text to be displayed as user transcript.

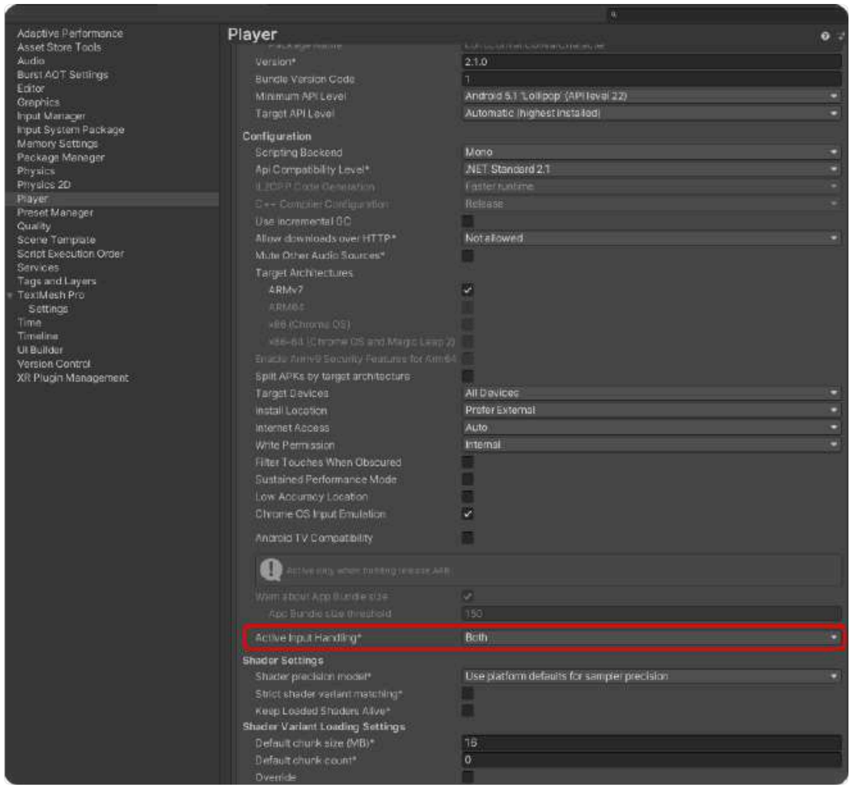
Input Management

Input Management - Efficiently handle input for Convai's Unity plugin integration.

Make sure that **Active Input Handling** in

"*Project Settings > Player*" is set to **Both** or **Input System Package (New)**.

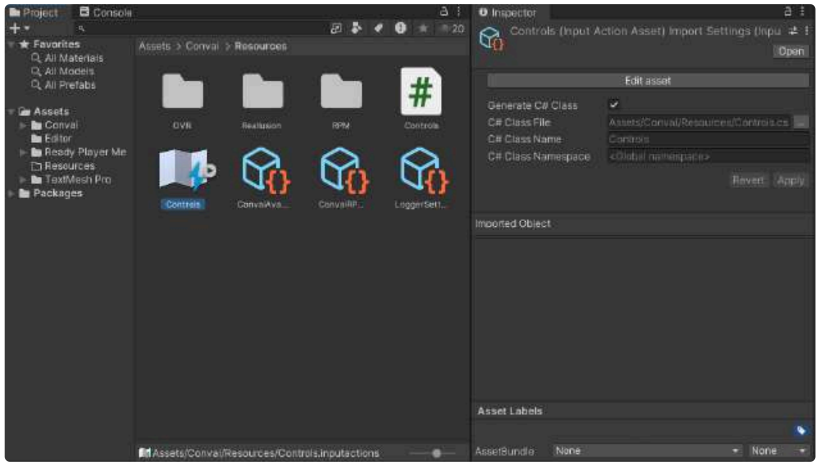
Our recommendation is Both. This way, you can use both the new and old input systems. Using the old input system can be faster when creating inputs for testing purposes.



Version* 2.1.0
Bundle Version Code 1
Minimum API Level Android 5.1 'Lollipop' (API level 22)
Target API Level Automatic (highest installed)
Configuration
Scripting Backend Mono
Api Compatibility Level* .NET Standard 2.1
IL2CPP Code Generation Faster runtime
C++ Compiler Configuration Release
Use incremental GC
Allow downloads over HTTP* Not allowed
Mute Other Audio Sources*
Target Architectures
ARMv7 ✓
ARM64
x86 (Chrome OS)
x86-64 (Chrome OS and Magic Leap 2)
Enable Armv9 Security Features for Arm64
Split APKs by target architecture
Target Devices All Devices
Install Location Prefer External
Internet Access Auto
Write Permission Internal
Filter Touches When Obscured
Sustained Performance Mode
Low Accuracy Location
Chrome OS Input Emulation ✓
Android TV Compatibility
Warn about App Bundle size ✓
App Bundle size threshold 150
Active Input Handling* Both
Shader Settings
Shader precision model* Use platform defaults for sampler precision
Strict shader variant matching*
Keep Loaded Shaders Alive*
Shader Variant Loading Settings
Default chunk size (MB)* 16
Default chunk count* 0
Override

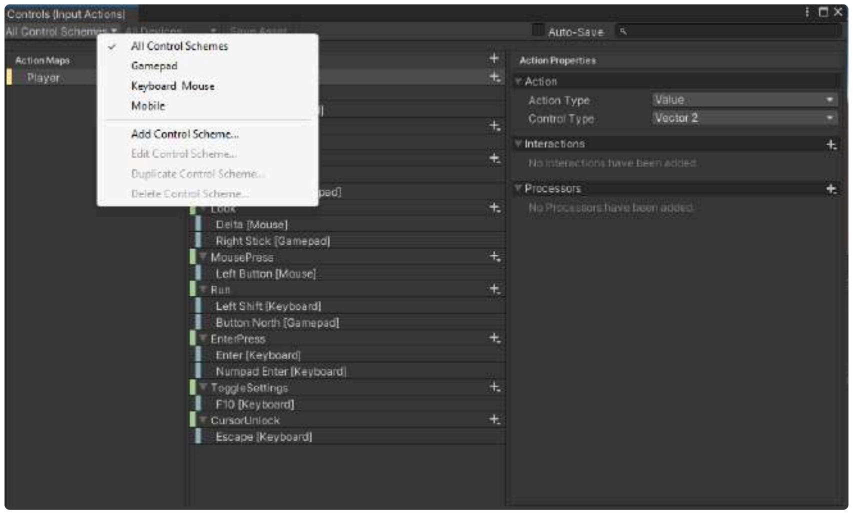
How to Change the Talk Button or Any Input?

1. Double click on the "**Controls**" asset in your project tab.

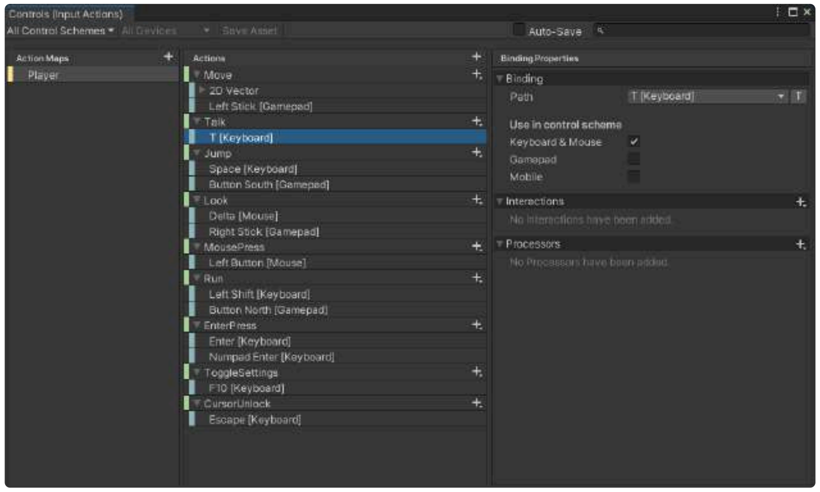


Controls (Input Action Asset) Import Settings
Generate C# Class ✓
C# Class File Assets/Convai/Resources/Controls.cs
C# Class Name Controls
C# Class Namespace <Global namespace>
Assets/Convai/Resources/Controls.inputactions

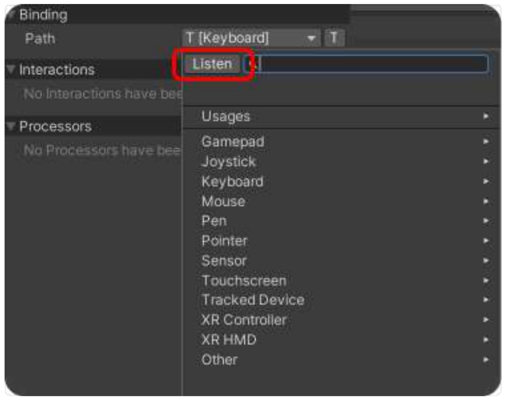
2. You can setup multiple control schemes for different devices here, currently we have it for PC (Keyboard & Mouse) and Gamepad. For mobile, we have provided joystick and buttons, which are mapped to Gamepad controls for functionality, but you can directly add touchscreen and use its different features to trigger an Input Action. You can also add your own control scheme if you want support for a different device by clicking on "Add Control Scheme".



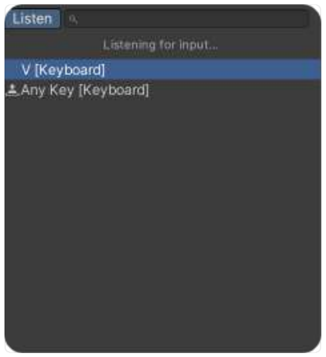
3. Find the Input Action you want to change in the above window. If you want to add a new Input Action, refer to the other section in documentation. In this case, we selected "**Talk Key Action**" to change the talk button. Click on "**T [Keyboard]**". In the Binding Properties window, click on the " **T [Keyboard]** " button in the Path field.



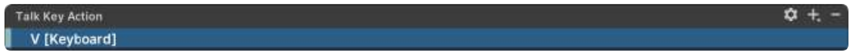
4. Press the " **Listen** " button in the top left of the opened window. If you prefer, you can choose your desired input from the categories below.



6. Press the key you want to assign.



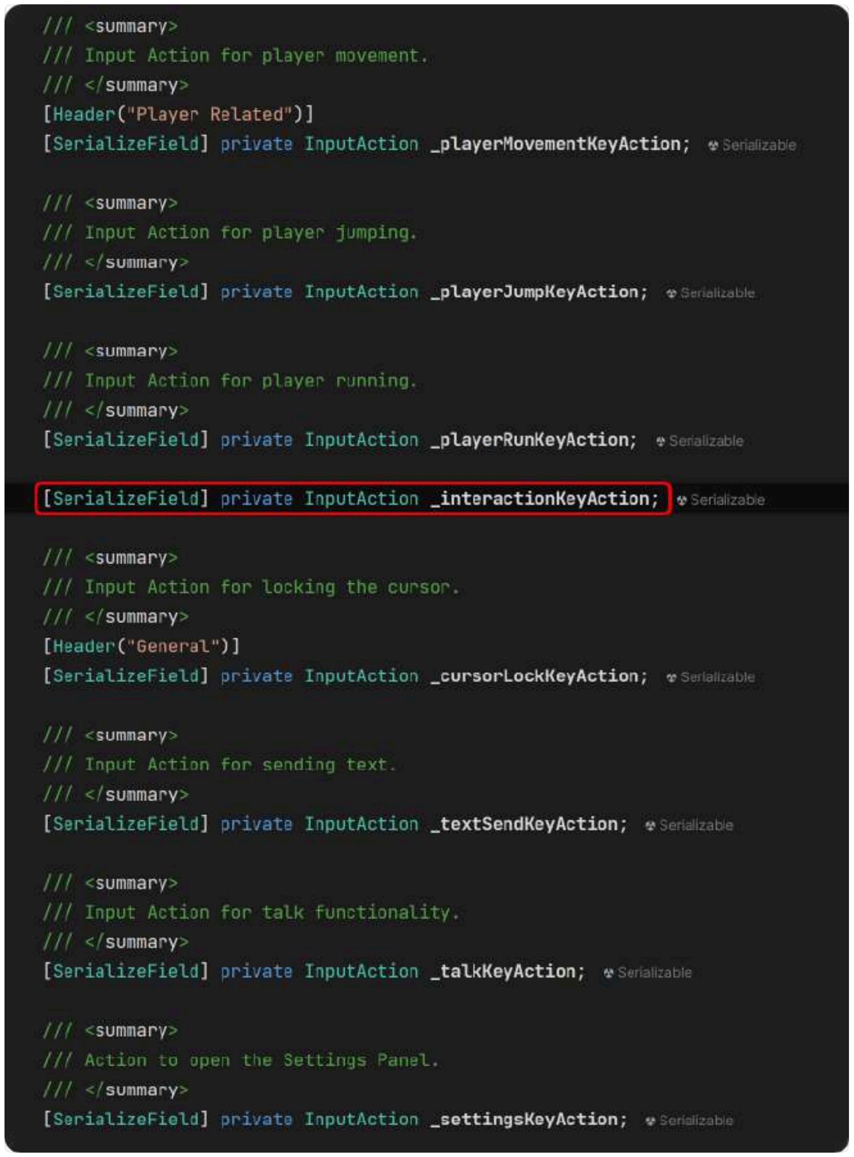
7. After assigning the desired key, you can see in the Inspector that the input for " **Talk Key Action** " has changed.



How to Add a New Input Action?

1. First, open the " **ConvaiInputManager.cs** " script.
   (" *Convai / Scripts / Utils / ConvaiInputManager.cs* ")
2. Create a variable of type **InputAction** in the field where other InputAction variables are located. We'll create an Interaction Key Action for example.

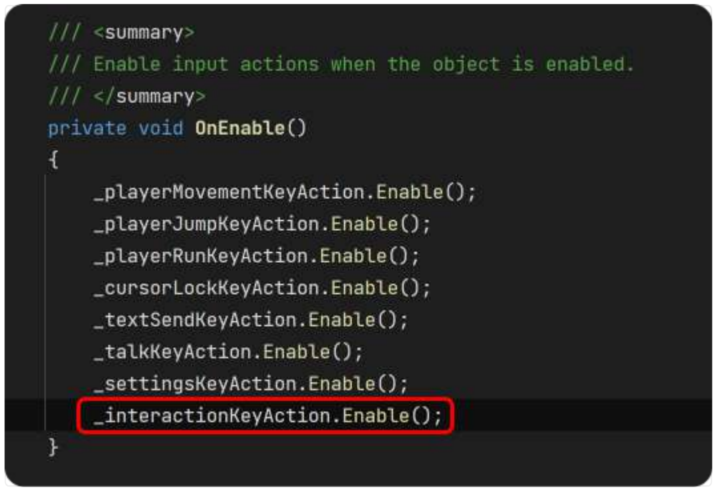
```
public InputAction InteractionKeyAction;
```



```
/// <summary>
/// Input Action for player movement.
/// </summary>
[Header("Player Related")]
[SerializeField] private InputAction _playerMovementKeyAction;

/// <summary>
/// Input Action for player jumping.
/// </summary>
[SerializeField] private InputAction _playerJumpKeyAction;

/// <summary>
/// Input Action for player running.
/// </summary>
[SerializeField] private InputAction _playerRunKeyAction;

[SerializeField] private InputAction _interactionKeyAction;

/// <summary>
/// Input Action for locking the cursor.
/// </summary>
[Header("General")]
[SerializeField] private InputAction _cursorLockKeyAction;

/// <summary>
/// Input Action for sending text.
/// </summary>
[SerializeField] private InputAction _textSendKeyAction;

/// <summary>
/// Input Action for talk functionality.
/// </summary>
[SerializeField] private InputAction _talkKeyAction;

/// <summary>
/// Action to open the Settings Panel.
/// </summary>
[SerializeField] private InputAction _settingsKeyAction;
```

3. In the OnEnable method, you need to enable the InputAction you created.

```
InteractionKeyAction.Enable();
```



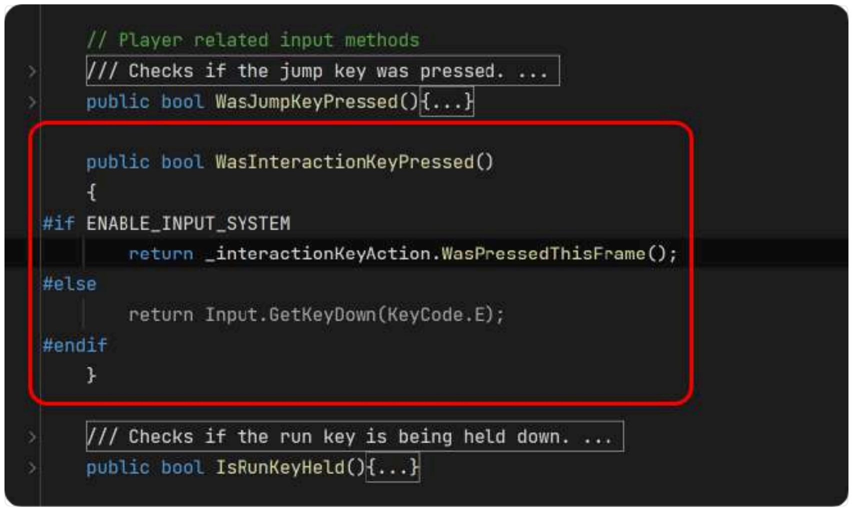
```
/// <summary>
/// Enable input actions when the object is enabled.
/// </summary>
private void OnEnable()
{
    _playerMovementKeyAction.Enable();
    _playerJumpKeyAction.Enable();
    _playerRunKeyAction.Enable();
    _cursorLockKeyAction.Enable();
    _textSendKeyAction.Enable();
    _talkKeyAction.Enable();
    _settingsKeyAction.Enable();
    _interactionKeyAction.Enable();
}
```

4. Now, let's create a method to access this InputAction.

```
public bool WasInteractionKeyPressed()
{
#if ENABLE_INPUT_SYSTEM
    return InteractionKeyAction.WasPressedThisFrame();
#else
    return Input.GetKeyDown(KeyCode.E);
#endif
}
```

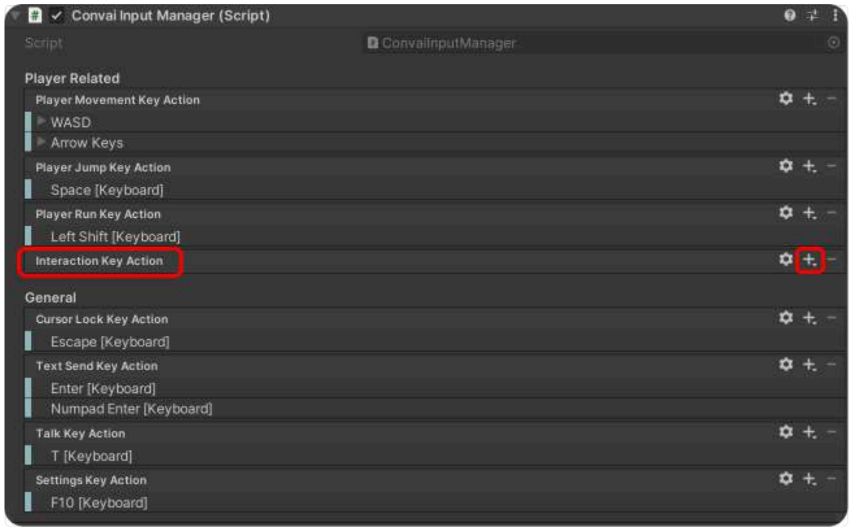
The **#if ENABLE_INPUT_SYSTEM** preprocessor directive ensures that this #if statement will function correctly when you select either **' Both '** or ' **Input System Package (New)** ' in " *Project Settings > Player > Active Input Handling* "

If you select " **Input Manager (Old)** " for Active Input Handling, the **#else** statement will work. Thus, in the **#else** section, we added the equivalent for the old input system.



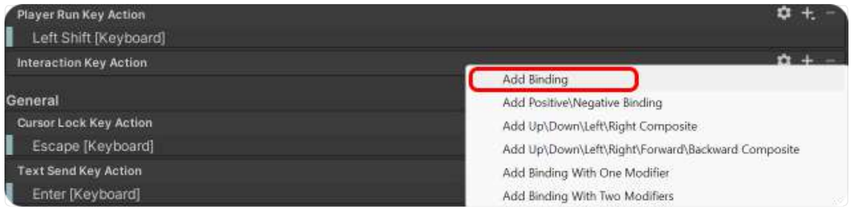
```
// Player related input methods
/// Checks if the jump key was pressed. ...
public bool WasJumpKeyPressed(){...}

public bool WasInteractionKeyPressed()
{
#if ENABLE_INPUT_SYSTEM
    return _interactionKeyAction.WasPressedThisFrame();
#else
    return Input.GetKeyDown(KeyCode.E);
#endif
}

/// Checks if the run key is being held down. ...
public bool IsRunKeyHeld(){...}
```

5. After applying the above steps, save by pressing " **CTRL+S** " Make sure you've done everything correctly.
6. Return to Unity, click on ConvaiInputManager and in the Inspector, you can see the added Input Action. To assign a key to this Input Action, click the " **+** " button.

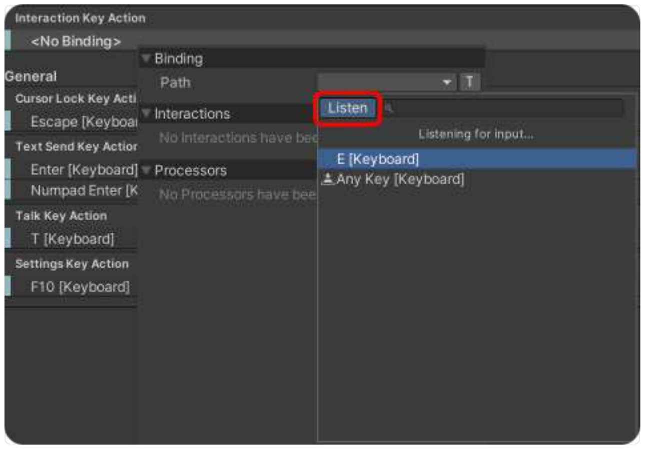


Convai Input Manager (Script)
Script ConvaiInputManager
Player Related
Player Movement Key Action
WASD
Arrow Keys
Player Jump Key Action
Space [Keyboard]
Player Run Key Action
Left Shift [Keyboard]
Interaction Key Action
General
Cursor Lock Key Action
Escape [Keyboard]
Text Send Key Action
Enter [Keyboard]
Numpad Enter [Keyboard]
Talk Key Action
T [Keyboard]
Settings Key Action
F10 [Keyboard]

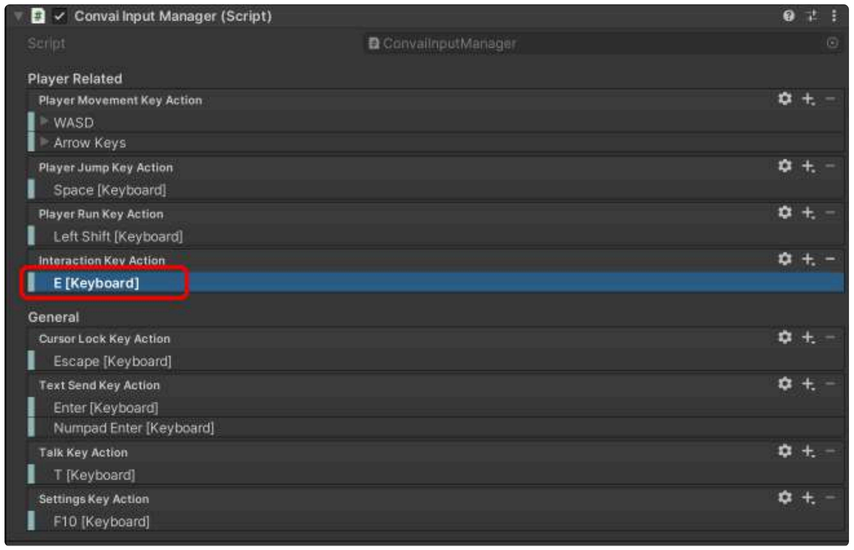
7. In the opened window, click on the " **Add Binding** " button.


Player Run Key Action
Left Shift [Keyboard]
Interaction Key Action
General
Cursor Lock Key Action
Escape [Keyboard]
Text Send Key Action
Enter [Keyboard]
Add Binding
Add Positive\Negative Binding
Add Up\Down\Left\Right Composite
Add Up\Down\Left\Right\Forward\Backward Composite
Add Binding With One Modifier
Add Binding With Two Modifiers

8. Click the " **Listen** " button in the opened window and press the key you want to assign or choose the desired key from the categories.



9. After selecting the desired key, you can see the assigned key under your Input Action in the Inspector.



Notification System

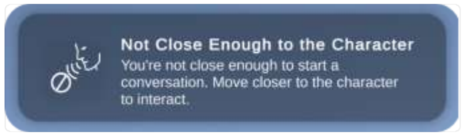
Notification System - Implement notifications with Convai Unity plugin utilities.

The Convai plugin comes with default notifications, totaling four. Here they are:

Notifications

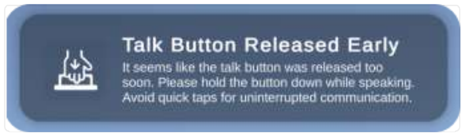
Not Close Enough to the Character

Appears when you press the talk button but there is no active NPC nearby.


Not Close Enough to the Character
You're not close enough to start a conversation. Move closer to the character to interact.

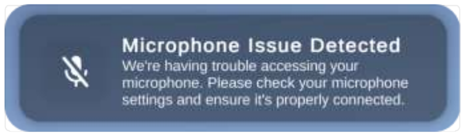
Talk Button Released Early

Appears if you release the talk button in less than 0.5 seconds.


Talk Button Released Early
It seems like the talk button was released too soon. Please hold the button down while speaking. Avoid quick taps for uninterrupted communication.

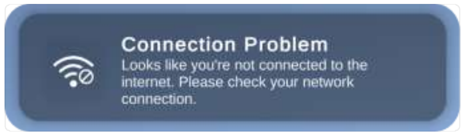
Microphone Issue Detected

Appears when the recorded audio input level is below the threshold.


Microphone Issue Detected
We're having trouble accessing your microphone. Please check your microphone settings and ensure it's properly connected.

Connection Problem

Appears when there is no internet connection upon launching the application.

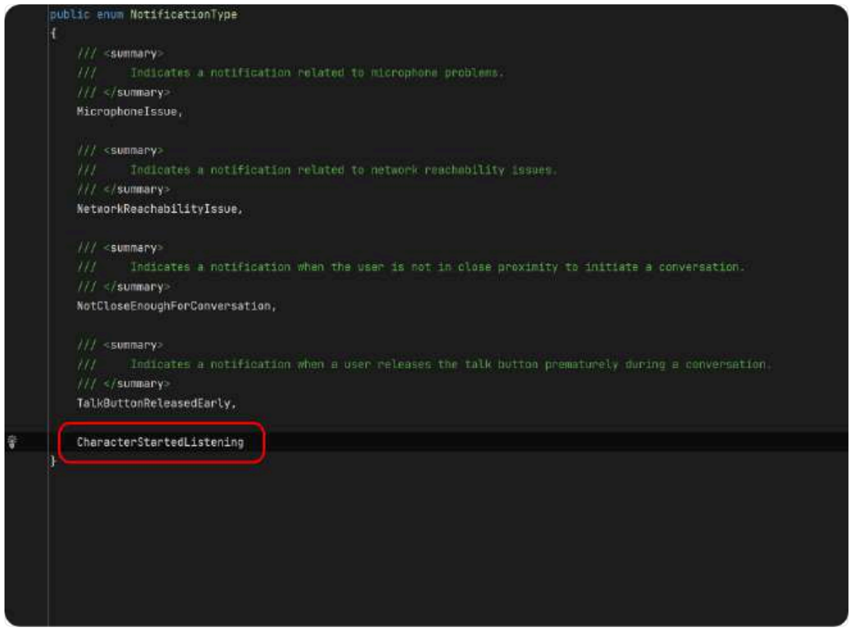

Connection Problem
Looks like you're not connected to the internet. Please check your network connection.

How to Add Your Own Notification?

Adding your custom notification is straightforward.

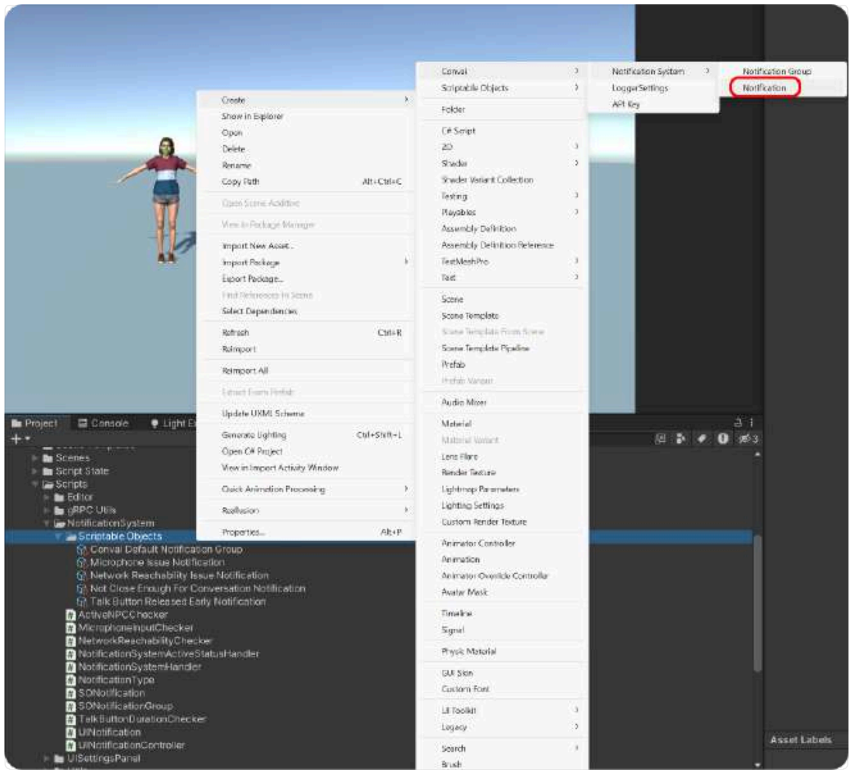
Let's go through the steps to add a " **CharacterStartedListening**" notification as an example.

1. Open the script *"Convai/Scripts/Notification System/Notification Type.cs."* This script stores Notification Types as enums. Give a name to your desired Notification type and add it here.

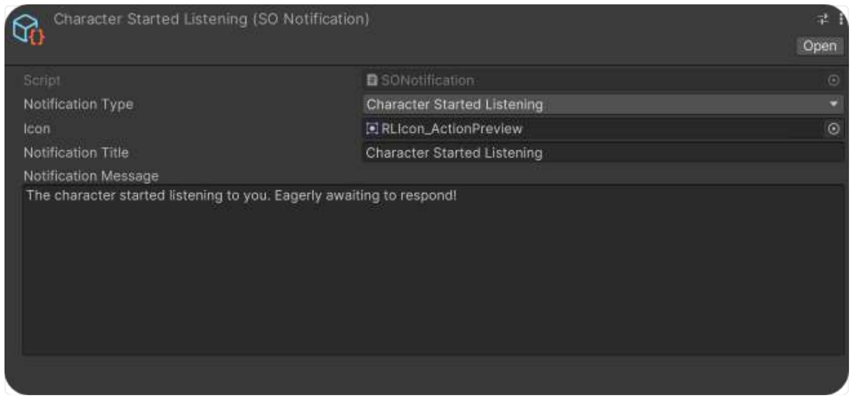


```
public enum NotificationType
{
    /// <summary>
    ///     Indicates a notification related to microphone problems.
    /// </summary>
    MicrophoneIssue,

    /// <summary>
    ///     Indicates a notification related to network reachability issues.
    /// </summary>
    NetworkReachabilityIssue,

    /// <summary>
    ///     Indicates a notification when the user is not in close proximity to initiate a conversation.
    /// </summary>
    NotCloseEnoughForConversation,

    /// <summary>
    ///     Indicates a notification when a user releases the talk button prematurely during a conversation.
    /// </summary>
    TalkButtonReleasedEarly,

    CharacterStartedListening
}
```

2. Right-click on *"Convai / Scripts / Notification System / Scriptable Objects"* and select *"Create > Convai > Notification System > Notification"* then create a **"Notification Scriptable Object".**



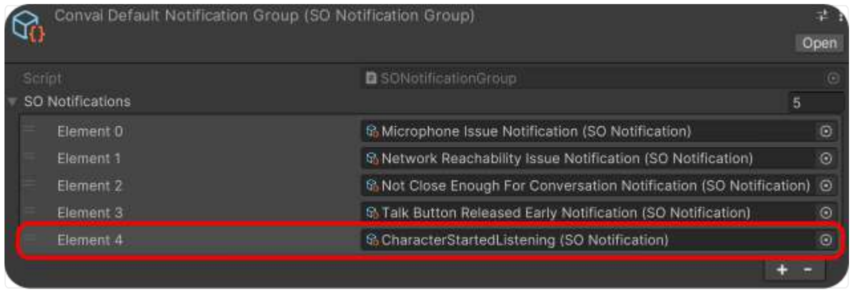
3. Name the created Notification Scriptable Object. Click on it, and fill in the fields in the Inspector as desired.


Character Started Listening (SO Notification)

Script	SONotification
Notification Type	Character Started Listening
Icon	RLIcon_ActionPreview
Notification Title	Character Started Listening
Notification Message	

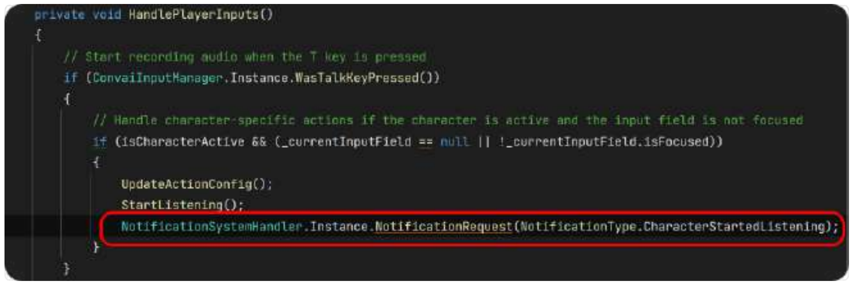
The character started listening to you. Eagerly awaiting to respond!

4. Add the created Notification Scriptable Object to *"Convai/Scripts/Notification System/Scriptable Objects"* under **"Convai Default Notification Group"** (details of Notification Group here****).


Convai Default Notification Group (SO Notification Group)

Script	SONotificationGroup
SO Notifications	5
Element 0	Microphone Issue Notification (SO Notification)
Element 1	Network Reachability Issue Notification (SO Notification)
Element 2	Not Close Enough For Conversation Notification (SO Notification)
Element 3	Talk Button Released Early Notification (SO Notification)
Element 4	CharacterStartedListening (SO Notification)

5. Your Notification is now ready. The last step is to call this Notification from where you need it. For example, if you created the " **CharacterStartedTalking** " Notification, find the location where your character listens and write the code.


```
private void HandlePlayerInputs()
{
    // Start recording audio when the T key is pressed
    if (ConvaiInputManager.Instance.WasTalkKeyPressed())
    {
        // Handle character-specific actions if the character is active and the input field is not focused
        if (isCharacterActive && (_currentInputField == null || !_currentInputField.isFocused))
        {
            UpdateActionConfig();
            StartListening();
            NotificationSystemHandler.Instance.NotificationRequest(NotificationType.CharacterStartedListening);
        }
    }
```

```
NotificationSystemHandler.Instance.NotificationRequest(NotificationType.Cl
```

6. Replace the parameter with the NotificationType you created. (For our example, ***NotificationType.CharacterStartedListening***)
7. Ensure that the Convai Notification System is present in your scene. (accessible from "*Convai/Prefabs/ Notification System*")

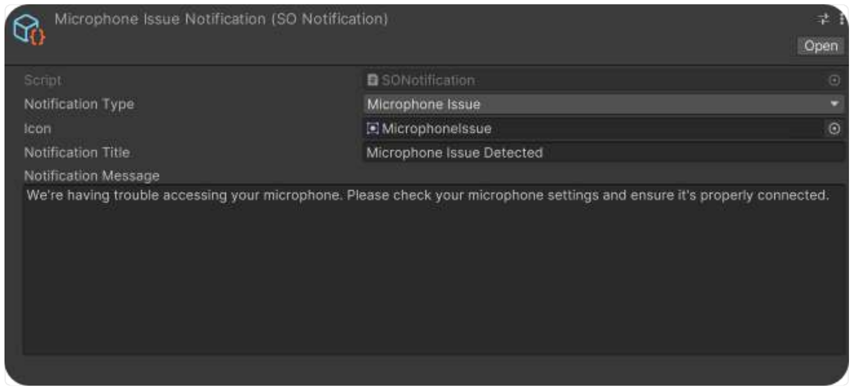
All steps are complete, and you're ready to test!🙂 ✅

Notification Scriptable Object

This Scriptable Object stores information about a Notification

- Notification Type
- Notification Icon
- Notification Title
- Notification Message

To create a new Notification Scriptable Object, right-click anywhere in the Project Window and select "*Create > Convai > Notification System > Notification*"


Microphone Issue Notification (SO Notification)

Script	SONotification
Notification Type	Microphone Issue
Icon	MicrophoneIssue
Notification Title	Microphone Issue Detected
Notification Message	

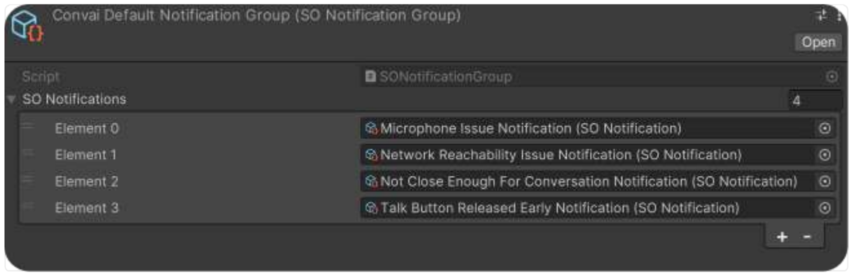
We're having trouble accessing your microphone. Please check your microphone settings and ensure it's properly connected.

Notification Group Scriptable Object

This Scriptable Object stores Notification Scriptable Objects as groups. When a Notification is requested, it searches for the Notification using the specified Notification Group in the Convai Notification System prefab's Notification System Handler script.

You can create different Notification groups based on your needs. Note: If your referenced Notification Group does not have the Notification you want, that Notification won't be called.

The Convai Default Notification Group has four Notifications, but you can add more or create a new group with additional notifications.



Convai Default Notification Group (SO Notification Group)

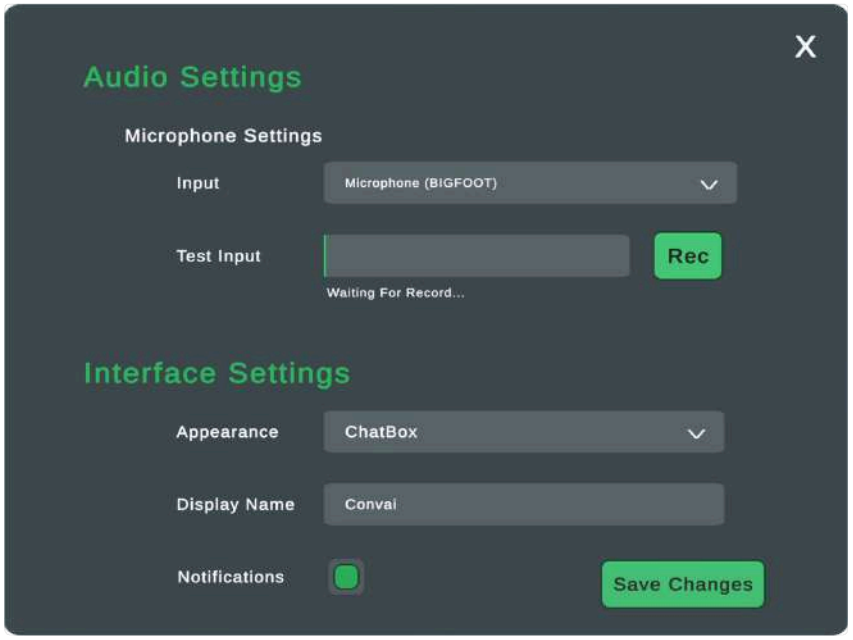
Script: SONotificationGroup

SO Notifications: 4

Element 0	Microphone Issue Notification (SO Notification)
Element 1	Network Reachability Issue Notification (SO Notification)
Element 2	Not Close Enough For Conversation Notification (SO Notification)
Element 3	Talk Button Released Early Notification (SO Notification)

Settings Panel

Settings Panel - Customize settings using Convai's Unity plugin utilities.

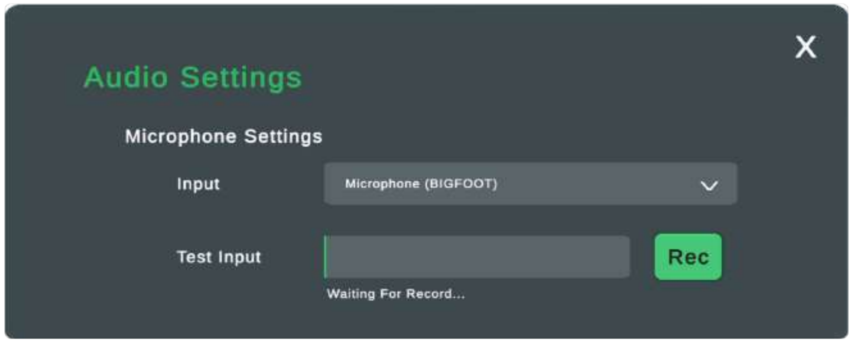


On the PC platform, you can open the Settings Panel by pressing F10.
For mobile platforms, you need to press the Settings button in the UI designs.

Settings Panel consists of two main sections.

- Audio Settings
- Interface Settings

Audio Settings

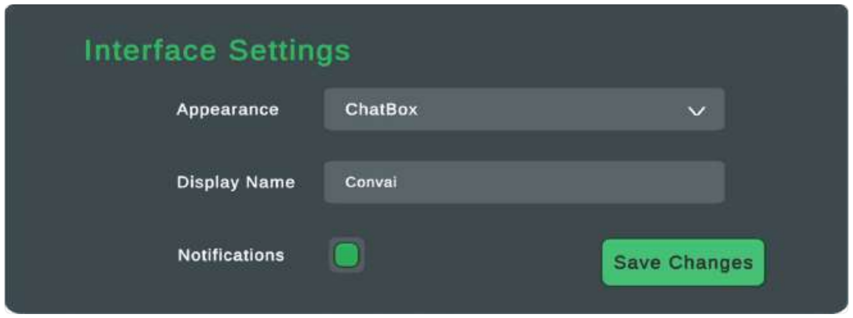


Microphone Settings

The Microphone Settings section is primarily for troubleshooting and testing the microphone when using the Convai plugin.

- In the Input section, you can view the microphones connected to your computer and select the desired one.
- In the Test Input field, you can record your voice using the selected microphone in the Input section. After clicking Stop, you can listen to the recorded voice and observe the sound levels.

Interface Settings



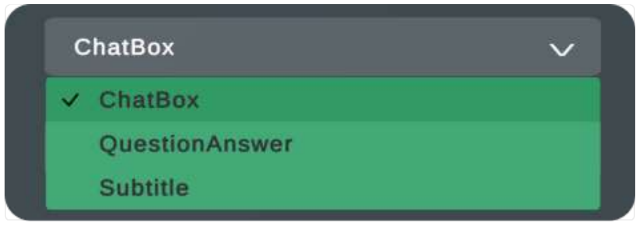
Appearance

The first setting that greets us here is the Appearance setting.

In the Appearance section, you can switch between Transcript UI designs.

There are three Transcript UI options:

- ChatBox
- QuestionAnswer
- Subtitle



Upon selecting a UI from the dropdown menu, you can preview it briefly.

Display Name

The second section in Interface Settings is the Display Name section. This section allows you to change how the user's name appears in the Transcript UI.

Notifications Checkmark

The last section in Interface Settings is the Notifications Checkmark.

Convai sometimes displays notifications on the screen to inform the user. If you want to disable these notifications, you can click the checkbox here. (If the box is green, it's active. If empty, it's inactive)

For more information about notifications, you can refer to this link.

Building for supported platforms

With Convai's Unity SDK, you can build your favorite application for several platforms, including Windows, MacOS and Android. Currently, we also support these platforms:

1. iOS/iPadOS
2. Virtual Reality (Occulus)
3. Augmented Reality (Android/iOS)
4. WebGL
5. Universal macOS applications

Building for iOS/iPadOS

This guide will walk you through the process of installing Convai-powered Unity applications on iOS and iPadOS devices.

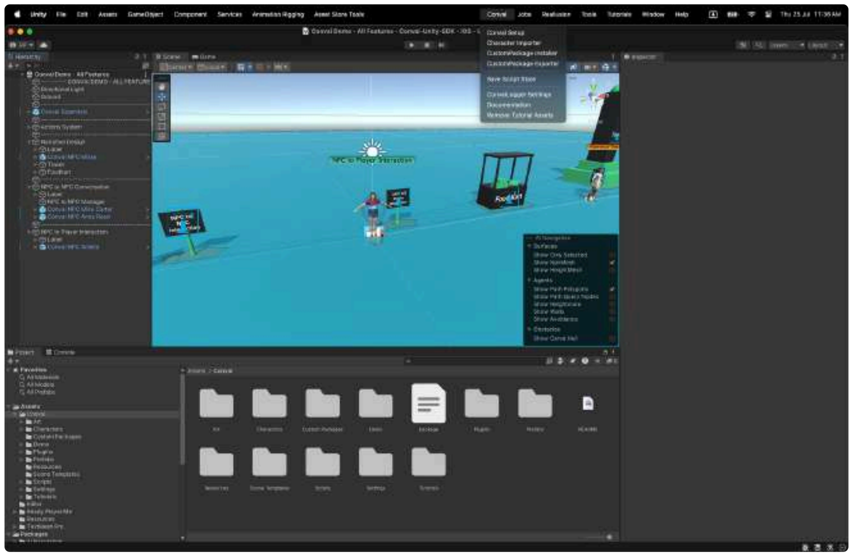
Prerequisites

Before you begin, make sure you have the following:

- Unity 2022.3 or later
- Xcode (latest version recommended)
- Apple Developer account
- Project with Convai's Unity SDK integrated and running properly
- MacBook for building and deploying to iOS/iPadOS

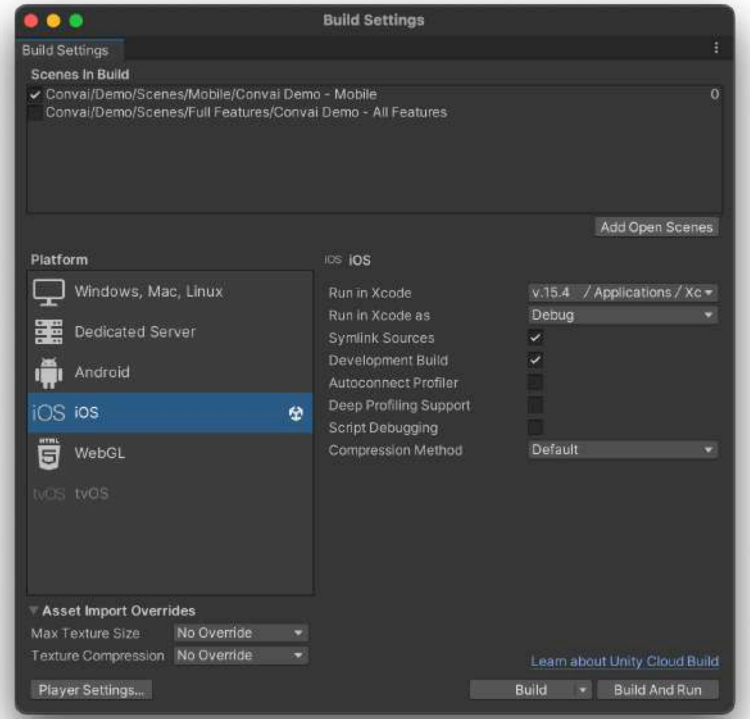
Step 1: Prepare Your Unity Project

1. Open your Convai-powered Unity project.
2. Ensure you have the latest version of the Convai Unity SDK imported and setup into your project.


Unity project with Convai SDK imported

Step 2: Configure Build Settings

1. In Unity, go to `File` → `Build Settings`.
2. Select `iOS` as the target platform.
3. Click `Switch Platform` if it's not already selected.
4. Check the `Development Build` option for testing purposes.


Build Settings

Scenes In Build
- [✓] Convai/Demo/Scenes/Mobile/Convai Demo - Mobile 0
- [ ] Convai/Demo/Scenes/Full Features/Convai Demo - All Features

Platform: Windows, Mac, Linux; Dedicated Server; Android; [✓ iOS]; WebGL; tvOS

Run in Xcode	v.15.4 / Applications / Xc
Run in Xcode as	Debug
Symlink Sources	✓
Development Build	✓
Autoconnect Profiler	
Deep Profiling Support	
Script Debugging	
Compression Method	Default

Asset Import Overrides

Max Texture Size	No Override
Texture Compression	No Override

Unity Build Settings window with iOS selected and Development Build checked

If you wish to add a few required files manually, follow step 3. If you want it to be done automatically, jump to step 4

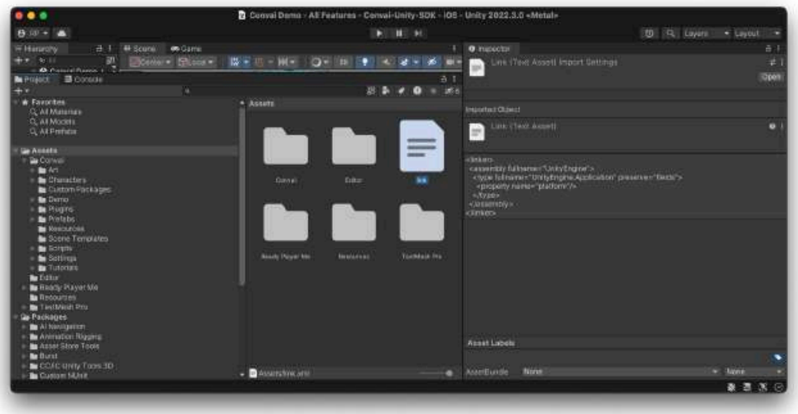
Step 3: Manually add Required Files

Add link.xml

1. Create a new file named `link.xml` in your project's `Assets` folder.

2. Add the following content to the file:

```xml
<linker>
  <assembly fullname="UnityEngine">
    <type fullname="UnityEngine.Application" preserve="fields">
      <property name="platform"/>
    </type>
  </assembly>
</linker>
```



```
<linker>
  <assembly fullname="UnityEngine">
    <type fullname="UnityEngine.Application" preserve="fields">
      <property name="platform"/>
    </type>
  </assembly>
</linker>
```

Unity project view showing the link.xml file in the Assets folder

This file prevents potential `FileNotFoundException` errors related to the `libgrpc_csharp_ext.x64.dylib` file.

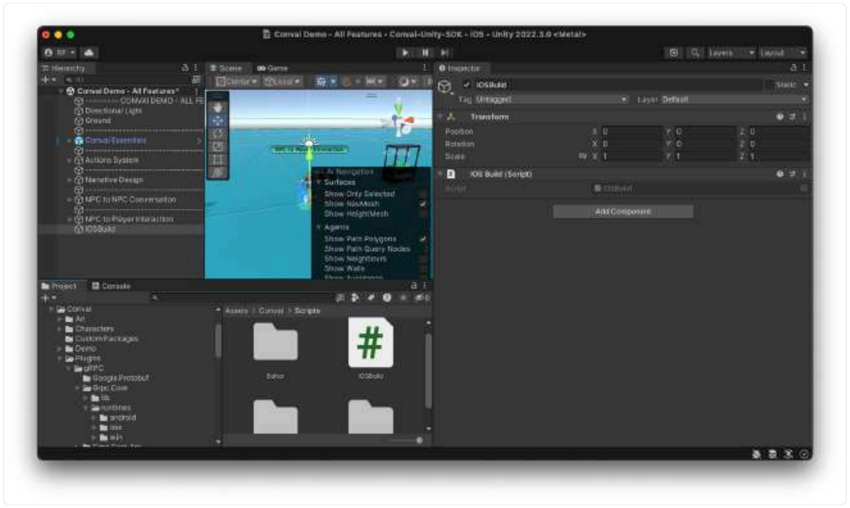
Add BuildIos.cs Script

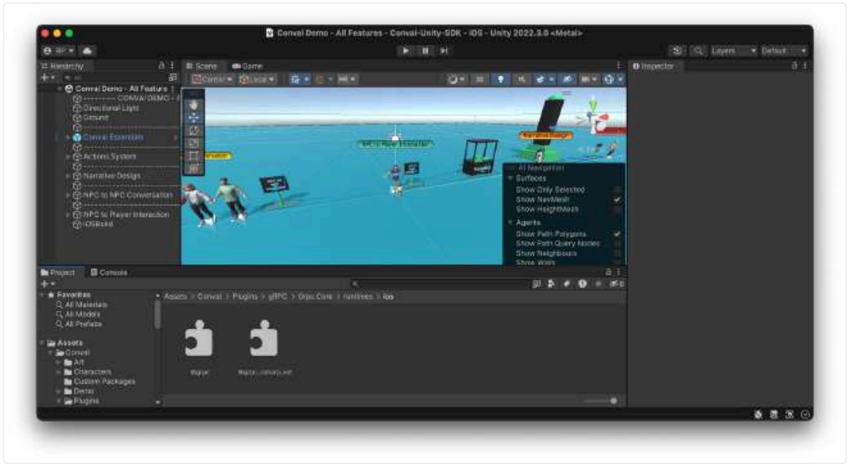
1. Create a new C# script in `Assets/Convai/Scripts` named `iOSBuild.cs`.
2. Add the following content to the script:

```csharp
#if UNITY_EDITOR && UNITY_IOS
using System.IO;
using UnityEditor;
using UnityEditor.Callbacks;
using UnityEditor.iOS.Xcode;
using UnityEngine;

public class iOSBuild : MonoBehaviour
{
    [PostProcessBuild]
    public static void OnPostProcessBuild(BuildTarget target, string path
    {
        string projectPath = PBXProject.GetPBXProjectPath(path);
        PBXProject project = new PBXProject();
        project.ReadFromString(File.ReadAllText(projectPath));
#if UNITY_2019_3_OR_NEWER
        string targetGuid = project.GetUnityFrameworkTargetGuid();
#else
        string targetGuid = project.TargetGuidByName(PBXProject.GetUnityT
#endif
        project.AddFrameworkToProject(targetGuid, "libz.tbd", false);
        project.SetBuildProperty(targetGuid, "ENABLE_BITCODE", "NO");
        File.WriteAllText(projectPath, project.WriteToString());
    }
}
#endif
```

Step 4: Install required gRPC dlls for iOS:

1. Go to Convai → Custom Package Installer
2. Click on `Install iOS Build Package`
3. Attach the script `iOSBuild.cs` to any GameObject in your scene.



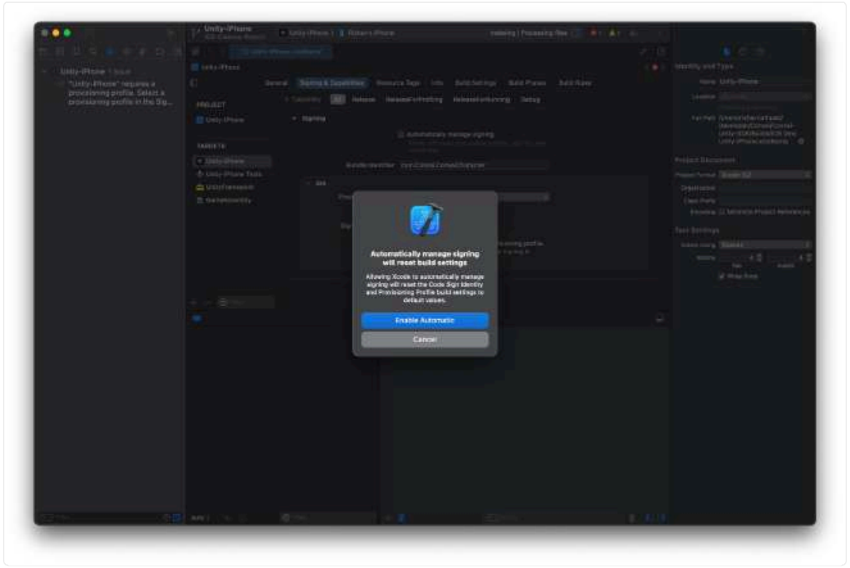


Step 5: Build the Xcode Project

1. In Unity, go to `File` → `Build Settings`.
2. Click `Build` and choose a location to save your Xcode project.
3. Wait for Unity to generate the Xcode project.

Step 6: Configure and Build in Xcode

1. Open the generated Xcode project.
2. In Xcode, select your project in the navigator.
3. Select your target under the "TARGETS" section.
4. Go to the "Signing & Capabilities" tab.
5. Ensure that "Automatically manage signing" is checked.
6. Select your Team from the dropdown (you need an Apple Developer account for this).
7. If needed, change the Bundle Identifier to a unique string.

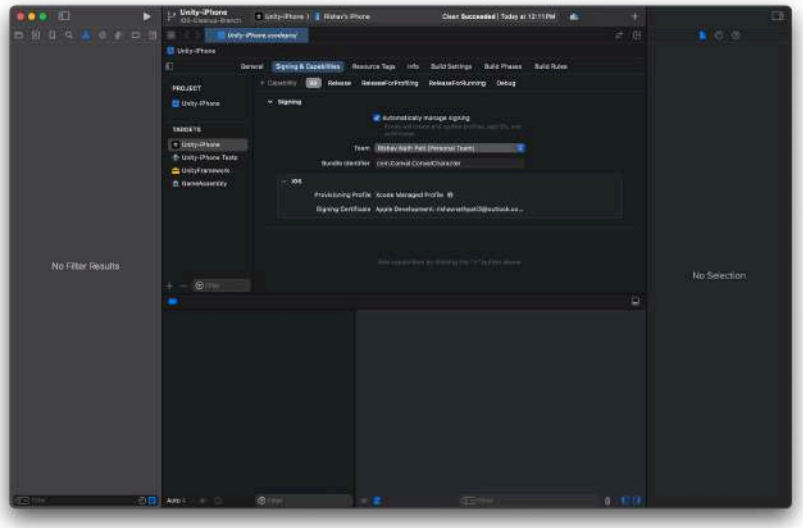


Xcode window showing the Signing & Capabilities tab with Team and Bundle Identifier fields highlighted

Step 7: Build and Run

1. Connect your iOS device to your Mac.

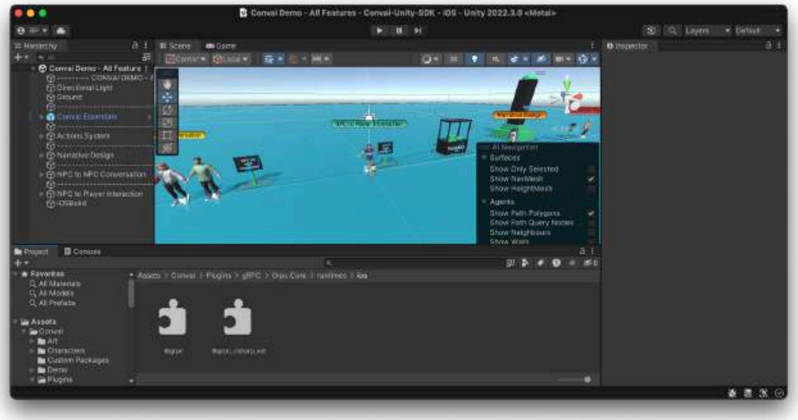
2. In Xcode, select your connected device as the build target.
3. Click the "Play" button or press `Cmd + R` to build and run the app on your device.



Xcode toolbar showing the connected device selected and the "Play" button highlighted

Troubleshooting

- If you encounter any build errors, ensure all the steps above have been followed correctly.
- Check that your Apple Developer account has the necessary provisioning profiles and certificates.
- If you face any GRPC-related issues, verify that the `libgrpc_csharp_ext.a` and `libgrpc.a` files are correctly placed in the `Assets/Convai/Plugins/gRPC/Grpc.Core/runtime/ios` folder.



Conclusion

You should now have successfully installed your Convai-powered Unity app on your iOS or iPadOS device. If you encounter any issues or need further assistance, please visit our [discord channel]() or contact Convai support.

Building for WebGL

This guide will help you integrate Convai's WebGL capabilities into your Unity projects, enabling you to bring to life AI characters with human-like conversational abilities.

Description

Convai's Unity WebGL SDK is designed to complement the standalone application capabilities of our Unity Asset Store version. With this specialized SDK, you can build and deploy interactive WebGL applications that leverage Convai's advanced conversation pipeline. Please see the instructions below or check out our *latest tutorial video* on YouTube.

Smart AI NPCs For Virtual Worlds Running in Your Web Browser | Co...

Getting Started

Download the SDK and Demos

Download the WebGL version of the Convai Unity SDK here.

Download the Complete WebGL Demo here.

Try out the Demo on itch.io here.

Prerequisites

Please ensure that Git is installed on your computer prior to proceeding. **Download Git from here.**

Import and Setup Instructions

Follow the Import and Setup Instructions from Import and Setup (nightly) and Setting Up Unity Plugin.

WebGL Incompatibility with Unity Editor

When attempting to play the scene in the Unity Editor, you may encounter the following error:

```
WebGL SDK does not run in Unity Editor. Please build and run in WebGL.
```

This error occurs because the WebGL SDK cannot be tested directly within the Unity Editor. To test your WebGL application, you must create a development build.

Switching to WebGL

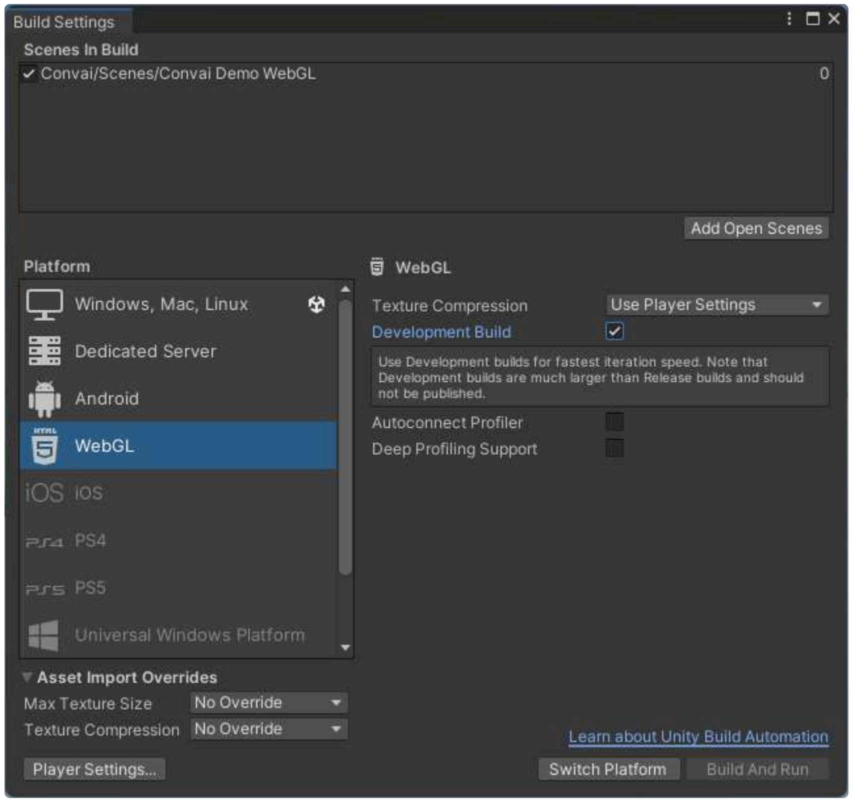
Now, your Unity setup is done, let's setup WebGL

Head on over to `File` → `Build Settings`, then:

1. Click on `WebGL`.
2. Check the `Development Build` box.
3. Select `Switch Platform.`

Patience, remember? This shift takes a bit.

After the platform is switched to WebGL, click on `Player Settings`.This is where the fun begins:



✓ Convai/Scenes/Convai Demo WebGL 0

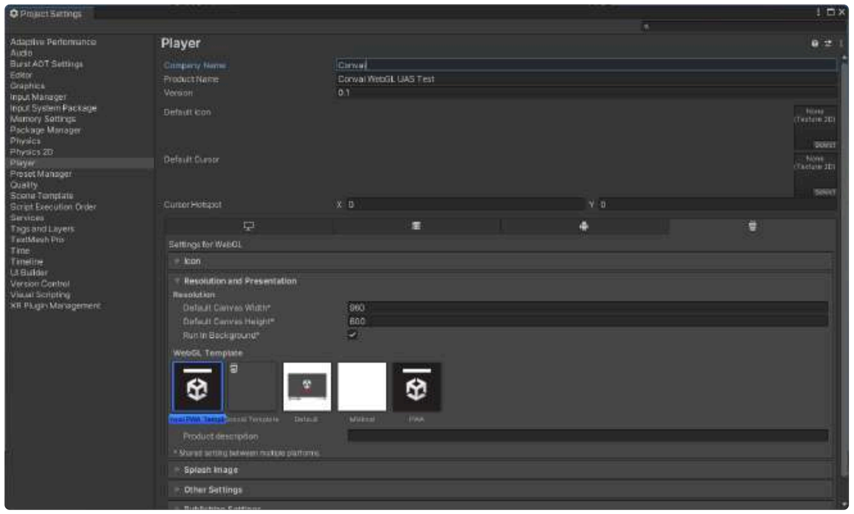
WebGL
Texture Compression: Use Player Settings
Development Build: ✓
Use Development builds for fastest iteration speed. Note that Development builds are much larger than Release builds and should not be published.
Autoconnect Profiler
Deep Profiling Support

Asset Import Overrides
Max Texture Size: No Override
Texture Compression: No Override

Configuring Player Settings

Once the platform conversion is complete,

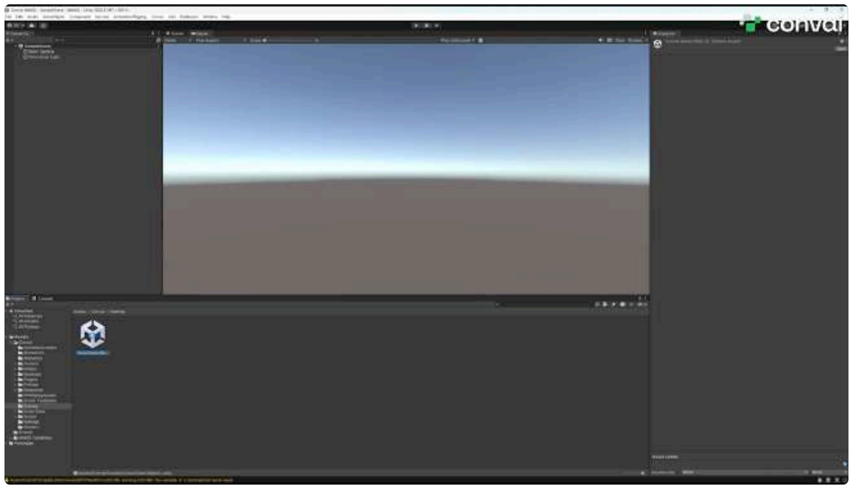
1. Open the file `Player Settings`.
2. Navigate to the page `WebGL settings`.
3. Under the `Resolution and Presentation` tab, select the `Convai PWA Template`.



Player
Company Name: Convai
Product Name: Convai WebGL UAS Test
Version: 0.1
Default Icon: None (Texture 2D)
Default Cursor: None (Texture 2D)
Cursor Hotspot: X 0 Y 0
Settings for WebGL
Resolution and Presentation
Resolution
Default Canvas Width*: 960
Default Canvas Height*: 600
Run In Background*: ✓
WebGL Template: [✓ Convai PWA Template] Convai Template, Default, Minimal, PWA
Product description
* Shared setting between multiple platforms.

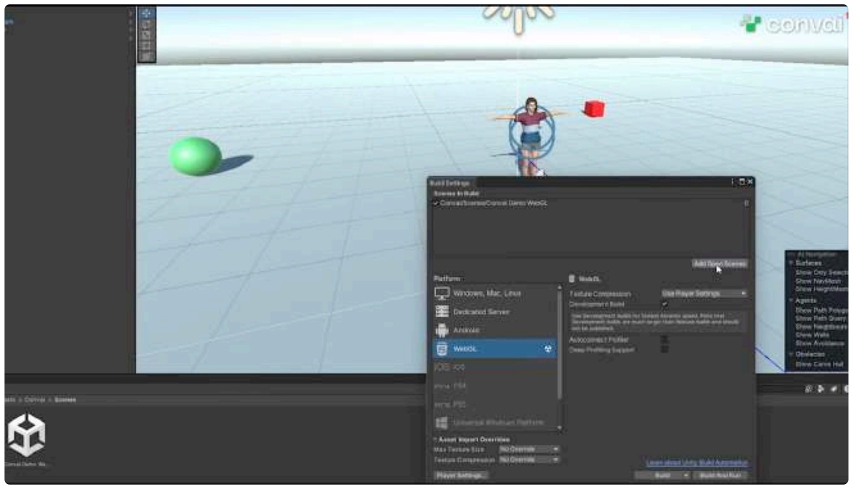
Importing Characters and Building the Scene

After the reloads is completed, check if the settings have changed. then, close all the open menus and follow these steps :



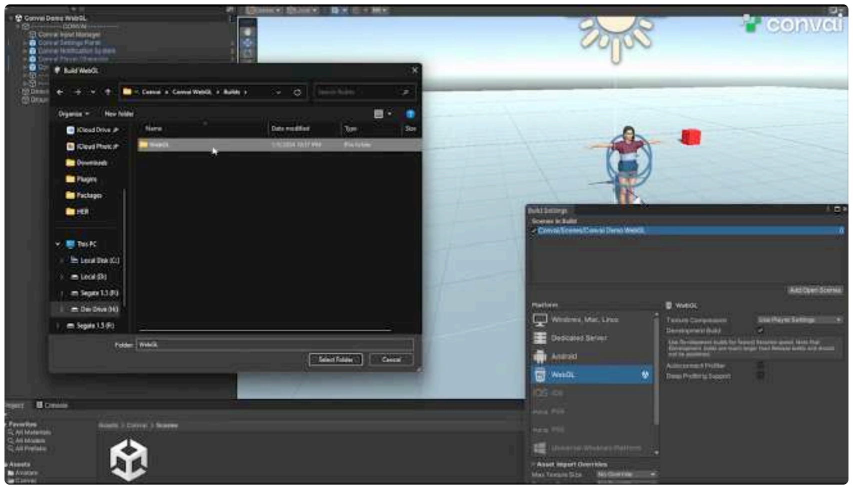
1. Double-click the `Convai folder` and go to scenes.
2. Open the `Convey Demo WebGL` scene.
3. Head to Convai's website, grab your API key, and input it back in Unity via `Convai` → `Convai Setup`.
4. Now, head again to the Convai's website and grab your favourite character's id and paste it to `Convai` → `Character Importer`.

Remember, Unity's editor won't let us test WebGL directly. But fear not, there's a `Build and Run` option:



1. Go back to `File` → `Build Settings`.
2. Click `Add Open Scenes` and then `Build and Run`.

Choose a folder for the build output, make a new one if needed, and name it "WebGL."

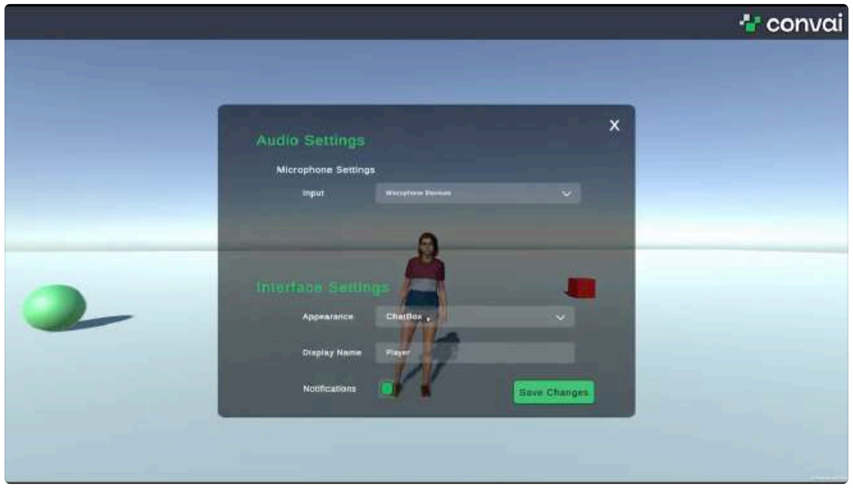


The First build may take some time.
For subsequent builds and runs, use the Unity shortcut key Ctrl + B.

The first build is the longest, so feel free to stretch a bit – but don't venture too far. Soon, you'll greet our demo character, Amelia, or any other character you brought into your digital oasis. Just give your Microphone permissions and here you go!

Engaging with your Convai AI NPCs

Now the magic happens. Press and hold 'T' to chat with your carefully cultivated character. Or click on the text box to type out a question. And for the attention to detail – press F10 to access the settings panel where you can change your name and the UI style to your liking.



Audio Settings

Microphone Settings

Input: Microphone Devices

Interface Settings

Appearance: ChatBox

Display Name: Player

Notifications: ✓

Feeling accomplished? You should! You now have a successfully working WebGL build in your browser. Curious developers can take a step further by downloading the project files from GitHub, available for all who desire to peek behind the curtain.

When you are ready with your production build, just uncheck the Development Build field in the Build Settings before publishing

Building for VR

Building for VR - Unity Plugin Guide for VR development with Convai.

VR Installation

If you want to make your Convai Plugin compatible with VR, you can do so using the automatic or manual process. Please see the instructions below or check out our *latest tutorial video* on YouTube.

Unleash the Power of Convai AI NPCs in Your Unity VR Game | Tutorial

Unleash the Power of Convai AI NPCs in Your Unity VR Game

Method 1 : Automatic Setup

Recommended for new projects.

The following processes will be performed:

- Universal Render Pipeline (URP)
- OpenXR Plugin
-

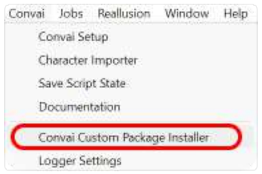
- XR Interaction Toolkit
- Convai Custom VR Package
- Convai URP Converter

If these packages are not present, they will be installed.

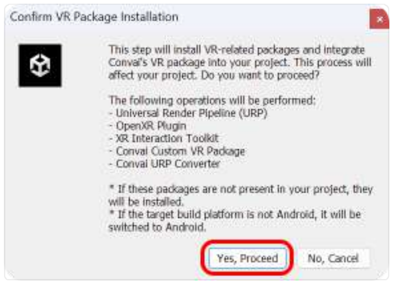
If the target build platform is not Android, it will be switched to *Android.*

Make sure to download the Android platform support from Unity Hub for your project's version.

1. Click on " *Convai / Convai Custom Package Installer / Install VR Package* "



2. Confirm the changes and processes to be made. If you agree, the process will start. Click " **Yes, Proceed** " and the process will begin. You'll see logs in the console.



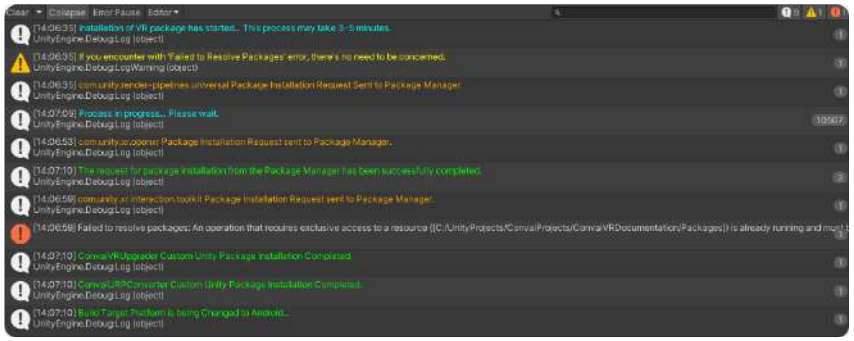
Confirm VR Package Installation

This step will install VR-related packages and integrate Convai's VR package into your project. This process will affect your project. Do you want to proceed?

The following operations will be performed:
- Universal Render Pipeline (URP)
- OpenXR Plugin
- XR Interaction Toolkit
- Convai Custom VR Package
- Convai URP Converter

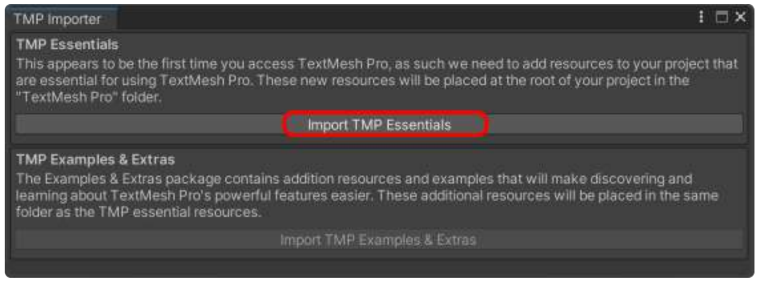
* If these packages are not present in your project, they will be installed.
* If the target build platform is not Android, it will be switched to Android.

3. If you encounter an error like " Failed to Resolve Packages " don't worry. The process will continue and the error will be resolved automatically after the package installations are complete.



```
[14:06:35] Installation of VR package has started... This process may take 3-5 minutes.
UnityEngine.Debug:Log (object)
[14:06:35] If you encounter with 'Failed to Resolve Packages' error, there's no need to be concerned.
UnityEngine.Debug:LogWarning (object)
[14:06:35] com.unity.render-pipelines.universal Package Installation Request Sent to Package Manager.
UnityEngine.Debug:Log (object)
[14:07:09] Process in progress... Please wait.
UnityEngine.Debug:Log (object)
[14:06:53] com.unity.xr.openxr Package Installation Request sent to Package Manager.
UnityEngine.Debug:Log (object)
[14:07:10] The request for package installation from the Package Manager has been successfully completed.
UnityEngine.Debug:Log (object)
[14:06:58] com.unity.xr.interaction.toolkit Package Installation Request sent to Package Manager.
UnityEngine.Debug:Log (object)
[14:06:58] Failed to resolve packages: An operation that requires exclusive access to a resource ([C:/UnityProjects/ConvaiProjects/ConvaiVRDocumentation/Packages]) is already running and must
[14:07:10] ConvaiVRUpgrader Custom Unity Package Installation Completed.
UnityEngine.Debug:Log (object)
[14:07:10] ConvaiURPConverter Custom Unity Package Installation Completed.
UnityEngine.Debug:Log (object)
[14:07:10] Build Target Platform is being Changed to Android...
UnityEngine.Debug:Log (object)
```

4. Open the " *Convai / Scenes / Convai Demo - VR* " demo scene. If the TMP Importer window appears, click " **Import TMP Essentials** " to install TextMeshPro for UI text objects.


TMP Importer

TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

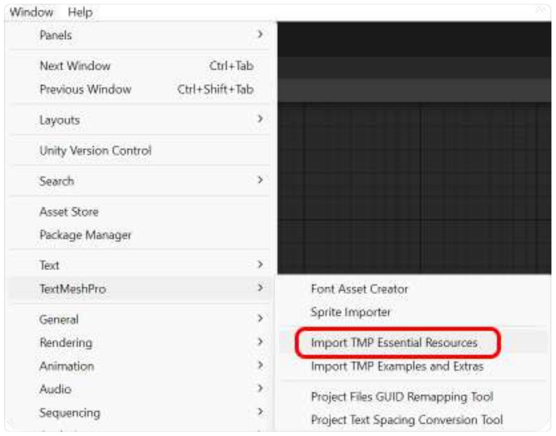
Import TMP Essentials

TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

Import TMP Examples & Extras

Alternatively, you can use the " *Window / TextMeshPro / Import TMP Essential Resources* " to install it.



5. Build your project by going to " *File /Build Settings / Build* " Ensure that the " **Convai Demo - VR** " scene is included in the Scenes in Build section.

Now everything is ready for testing. 🙂 ✅

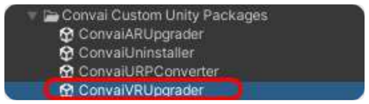
Ensure you've set up your API Key. (Convai / Convai Setup)

Method 2 : Manual Setup

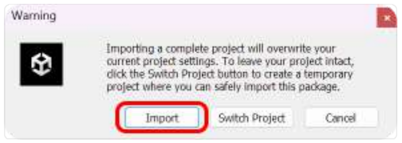
Ensure you have the following packages installed in your project:

- OpenXR or Oculus XR
- XR Interaction Toolkit
- URP (Universal Render Pipeline)

1. Double-click on " *Convai / Convai Custom Unity Packages / ConvaiVRUpgrader.unitypackage* "



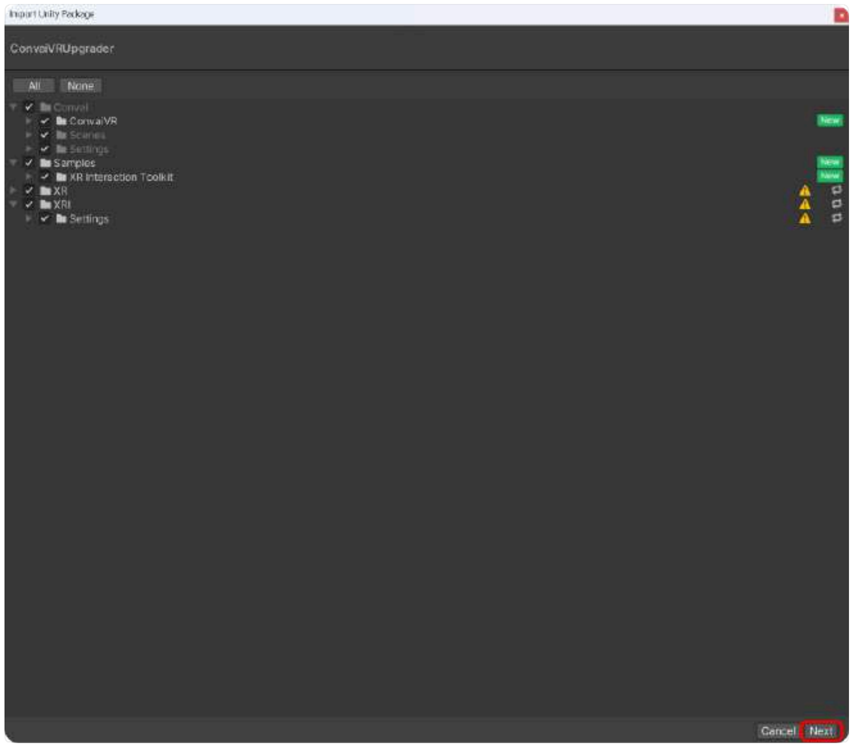
2. You'll see a warning that the settings will overwrite your project settings. You can either allow it by clicking " **Import** " or create a temporary project by clicking " **Switch Project** "



Warning

Importing a complete project will overwrite your current project settings. To leave your project intact, click the Switch Project button to create a temporary project where you can safely import this package.

3. In the Import Unity Package window, review the assets to be imported and click " **Next** "

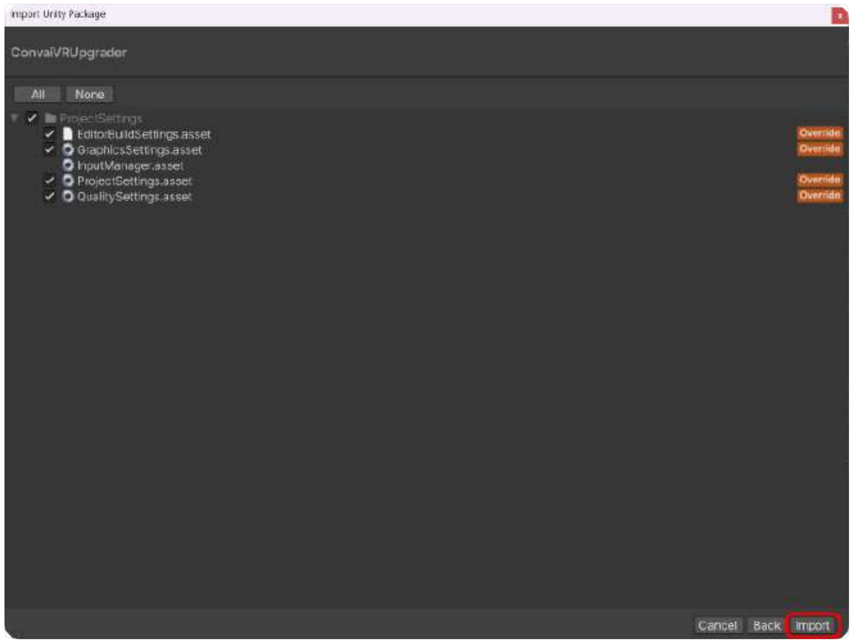


Import Unity Package

ConvaiVRUpgrader

```
✓ Convai
  ✓ ConvaiVR        New
  ✓ Scenes
  ✓ Settings
✓ Samples           New
  ✓ XR Interaction Toolkit   New
✓ XR
✓ XRI
  ✓ Settings
```

4. In this window, select the project settings you want to import and complete the installation by clicking " **Import** ".

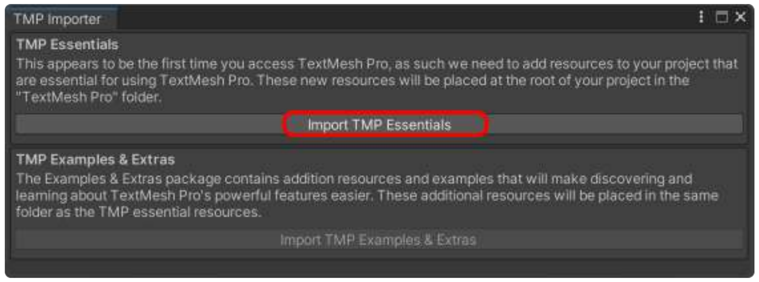


Import Unity Package

ConvaiVRUpgrader

```
✓ ProjectSettings
  ✓ EditorBuildSettings.asset   Override
  ✓ GraphicsSettings.asset      Override
    InputManager.asset
  ✓ ProjectSettings.asset       Override
  ✓ QualitySettings.asset       Override
```

5. Open the " *Convai / Scenes/ Convai Demo - VR* " demo scene. If the TMP Importer window appears, click " **Import TMP Essentials** " to install TextMeshPro for UI text objects.



TMP Importer

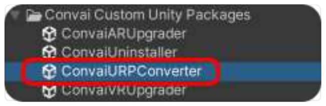
TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

TMP Examples & Extras

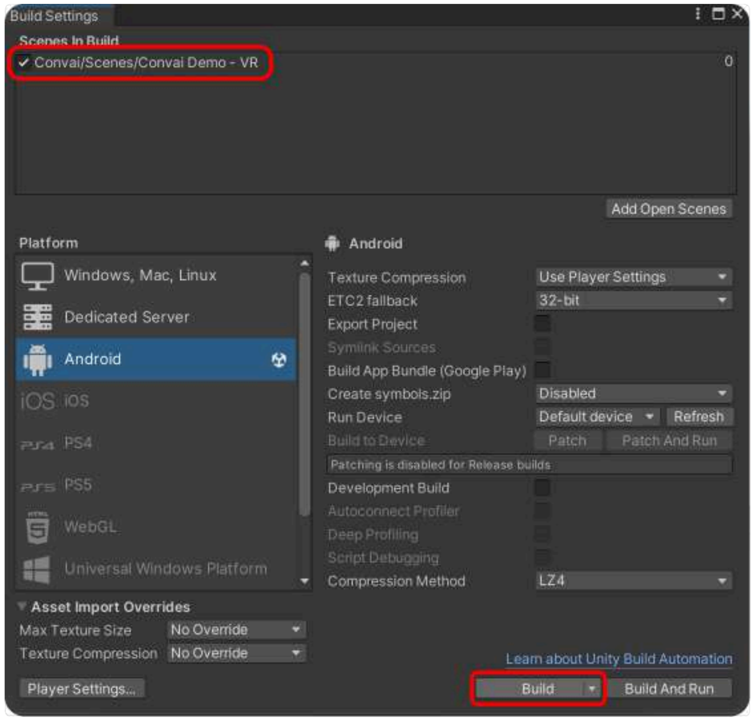
The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

6. If you see 3D objects in pink, it's a shader issue. If you're using URP, convert the materials to URP by double-clicking on " *Convai / Convai Custom Unity Packages / ConvaiURPConverter* " and importing all assets in the window that appears.



Convai Custom Unity Packages
ConvaiARUpgrader
ConvaiUninstaller
ConvaiURPConverter
ConvaiVRUpgrader

7. Build your project by going to " *File / Build Settings / Build* " Ensure that the " **Convai Demo - VR** " scene is included in the Scenes in **Build** section.



Build Settings

Scenes In Build
✓ Convai/Scenes/Convai Demo - VR 0

Add Open Scenes

Platform
Windows, Mac, Linux
Dedicated Server
Android
iOS
PS4
PS5
WebGL
Universal Windows Platform

Android

Texture Compression	Use Player Settings
ETC2 fallback	32-bit
Export Project	
Symlink Sources	
Build App Bundle (Google Play)	
Create symbols.zip	Disabled
Run Device	Default device · Refresh
Build to Device	Patch · Patch And Run
Patching is disabled for Release builds	
Development Build	
Autoconnect Profiler	
Deep Profiling	
Script Debugging	
Compression Method	LZ4

Asset Import Overrides

Max Texture Size	No Override
Texture Compression	No Override

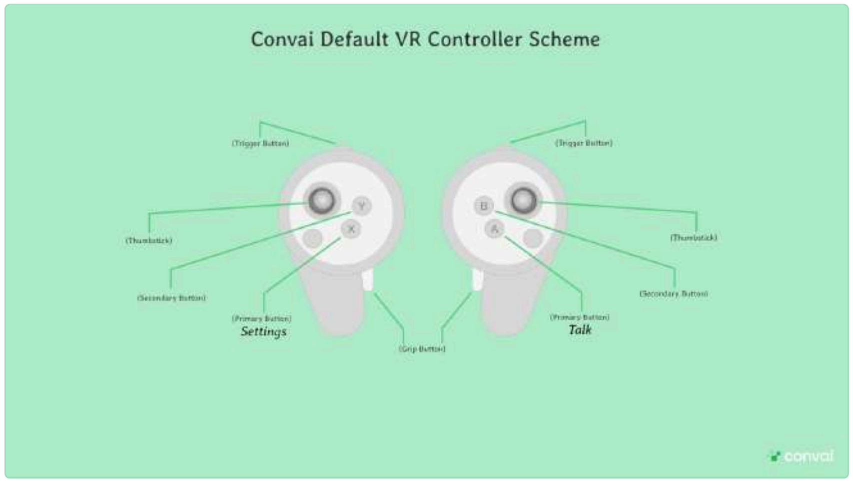
Learn about Unity Build Automation

Player Settings... Build Build And Run

Now everything is ready for testing. 🙂✅

ⓘ Ensure you've set up your API Key (Convai/Convai Setup).

Convai Default VR Controller Scheme

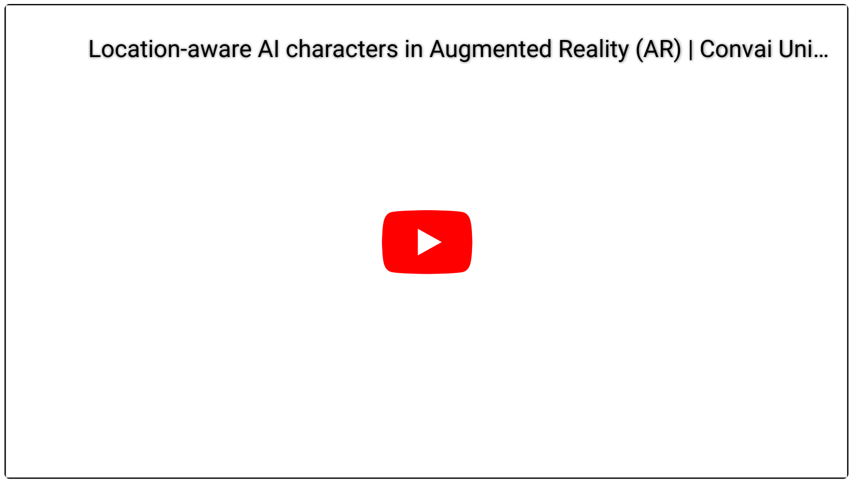


Building for AR

Building for AR - Unity Plugin Guide for AR development with Convai.

AR Installation

If you want to make your Convai Plugin compatible with AR, you can do so in two ways. Please see the instructions below or check out our *latest tutorial video* on YouTube.


Location-aware AI characters in Augmented Reality (AR) | Convai Uni...

Method 1 : Automatic Setup

Recommended for new projects.

The following processes will be performed:

- Universal Render Pipeline (URP)
- ARCore Plugin
-

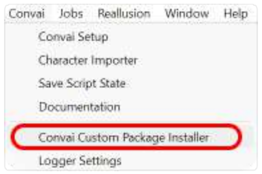
- Convai Custom AR Package
- Convai URP Converter

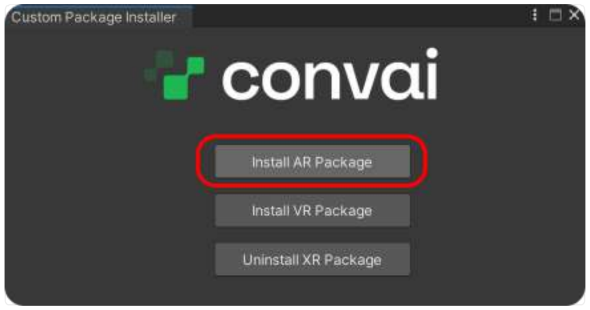
If these packages are not present, they will be installed.

If the target build platform is not Android, it will be switched to *Android.*

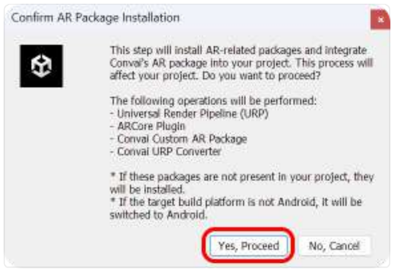
Make sure to download the Android platform support from Unity Hub for your project's version.

1. Click on " *Convai / Convai Custom Package Installer / Install AR Package* "





2. Confirm the changes and processes to be made. If you agree, the process will start. Click " **Yes, Proceed** " and the process will begin. You'll see logs in the console.

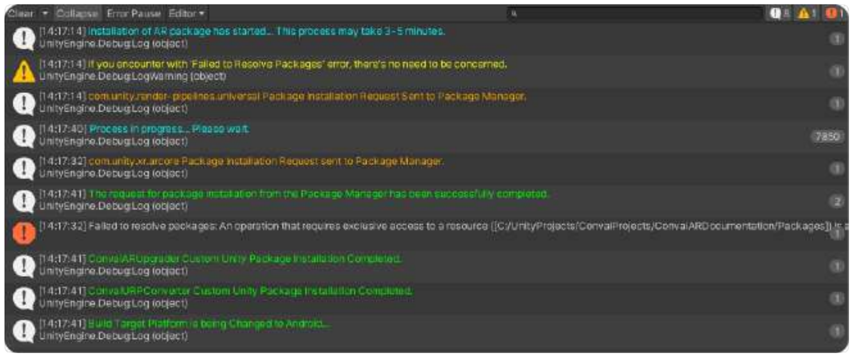

Confirm AR Package Installation

This step will install AR-related packages and integrate Convai's AR package into your project. This process will affect your project. Do you want to proceed?

The following operations will be performed:
- Universal Render Pipeline (URP)
- ARCore Plugin
- Convai Custom AR Package
- Convai URP Converter

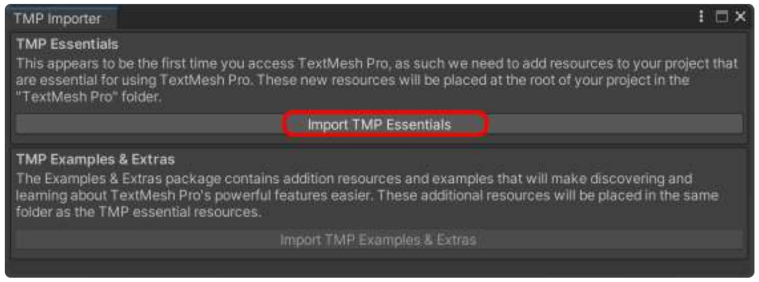
* If these packages are not present in your project, they will be installed.
* If the target build platform is not Android, it will be switched to Android.

3. If you encounter an error like "Failed to Resolve Packages," don't worry. The process will continue, and the error will be resolved automatically after the package installations are complete.



```
[14:17:14] Installation of AR package has started... This process may take 3-5 minutes.
UnityEngine.Debug:Log (object)
[14:17:14] If you encounter with 'Failed to Resolve Packages' error, there's no need to be concerned.
UnityEngine.Debug:LogWarning (object)
[14:17:14] com.unity.render-pipelines.universal Package Installation Request Sent to Package Manager.
UnityEngine.Debug:Log (object)
[14:17:40] Process in progress... Please wait.                                         7850
UnityEngine.Debug:Log (object)
[14:17:32] com.unity.xr.arcore Package Installation Request sent to Package Manager.
UnityEngine.Debug:Log (object)
[14:17:41] The request for package installation from the Package Manager has been successfully completed.   2
UnityEngine.Debug:Log (object)
[14:17:32] Failed to resolve packages: An operation that requires exclusive access to a resource ([C:/UnityProjects/ConvaiProjects/ConvaiARDocumentation/Packages])
[14:17:41] ConvaiARUpgrader Custom Unity Package Installation Completed.
UnityEngine.Debug:Log (object)
[14:17:41] ConvaiURPConverter Custom Unity Package Installation Completed.
UnityEngine.Debug:Log (object)
[14:17:41] Build Target Platform is being Changed to Android...
UnityEngine.Debug:Log (object)
```

4. Open the " *Convai / Scenes / Convai Demo - AR* " demo scene. If the TMP Importer window appears (It will appear if TMP Essentials is not installed in your project), click " **Import TMP Essentials** " to install TextMeshPro Essentials for UI text objects.


TMP Importer

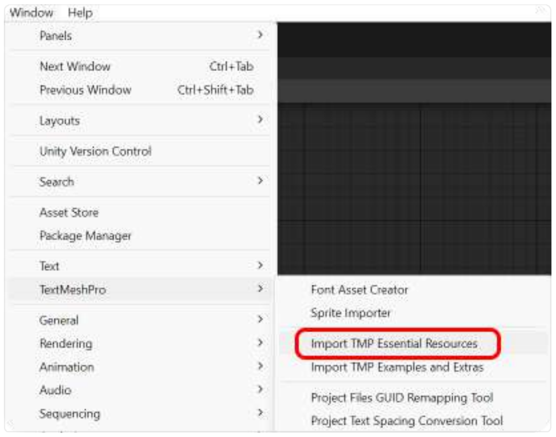
TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

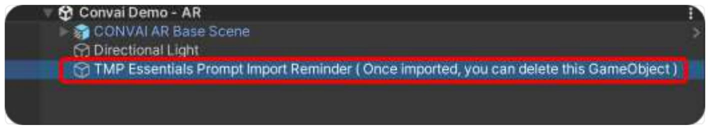
TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

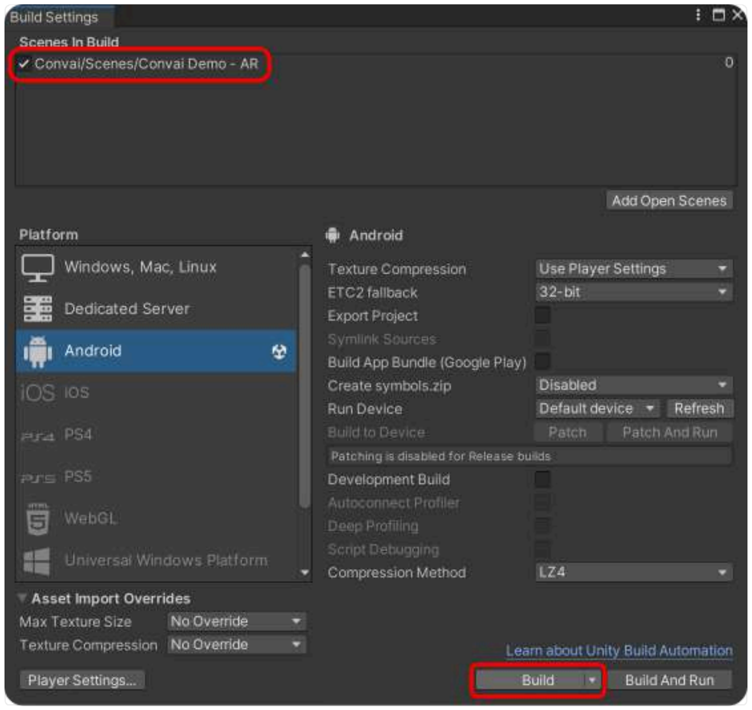
Alternatively, you can use the " *Window / TextMeshPro / Import TMP Essential Resources* " to install it.



5. After importing TMP Essentials, you can remove the empty GameObject in your scene that triggers the Prompt window to appear.


Convai Demo - AR
CONVAI AR Base Scene
Directional Light
TMP Essentials Prompt Import Reminder (Once imported, you can delete this GameObject)

6. Build your project by going to " *File / Build Settings / Build* " Ensure that the " **Convai Demo - AR** " scene is included in the Scenes in Build section.



✓ Convai/Scenes/Convai Demo - AR 0

Texture Compression: Use Player Settings
ETC2 fallback: 32-bit
Export Project
Symlink Sources
Build App Bundle (Google Play)
Create symbols.zip: Disabled
Run Device: Default device
Build to Device
Patching is disabled for Release builds
Development Build
Autoconnect Profiler
Deep Profiling
Script Debugging
Compression Method: LZ4

Asset Import Overrides
Max Texture Size: No Override
Texture Compression: No Override

Ensure you've set up your API Key. (Convai / Convai Setup)

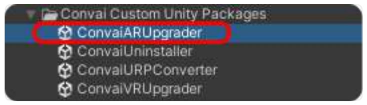
Now everything is ready for testing. 🙂✅

Method 2 : Manual Setup

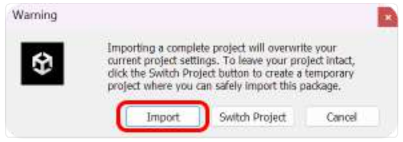
Ensure you have the following packages installed in your project:

- ARCore
- URP (Universal Render Pipeline) - Recommended for optimization, though not mandatory

1. Double-click on " Convai / Convai Custom Unity Packages / ConvaiVRUpgrader.unitypackage "

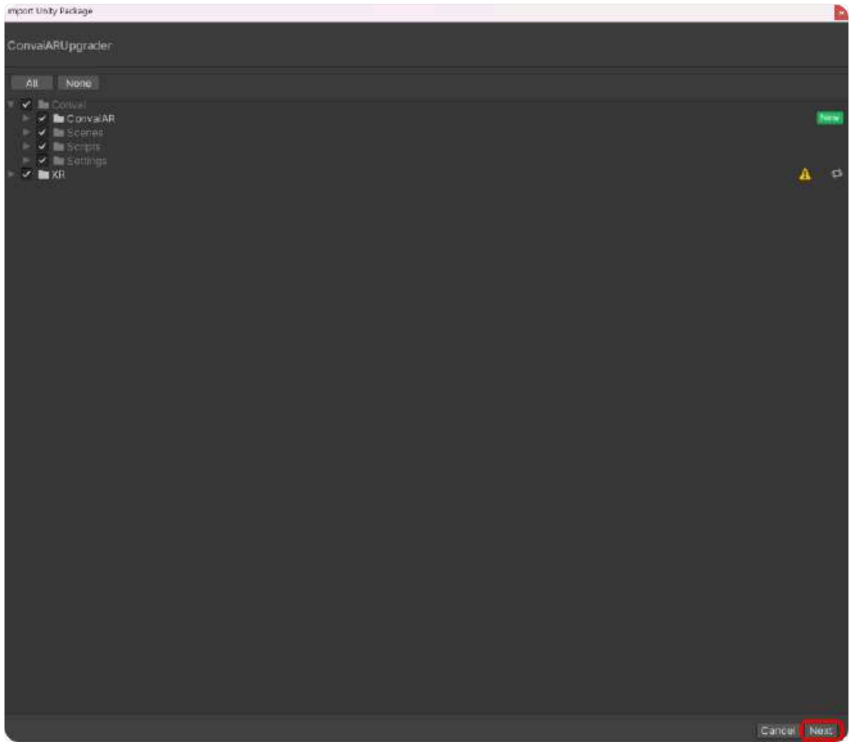


2. You'll see a warning that the settings will overwrite your project settings. You can either allow it by clicking " **Import** " or create a temporary project by clicking " **Switch Project** "



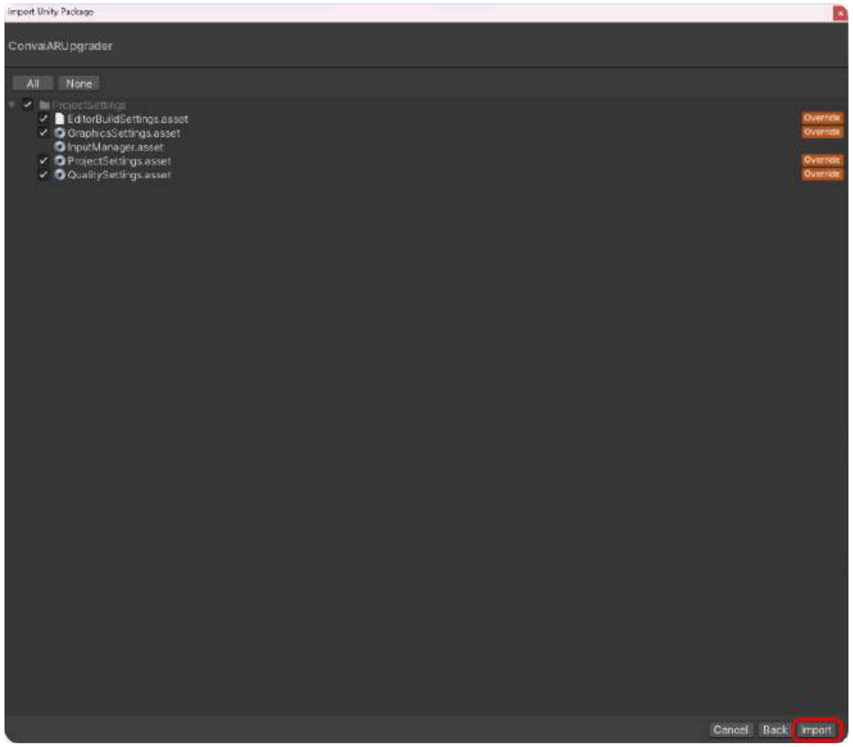
Importing a complete project will overwrite your current project settings. To leave your project intact, click the Switch Project button to create a temporary project where you can safely import this package.

3. In the Import Unity Package window, review the assets to be imported and click " **Next** "



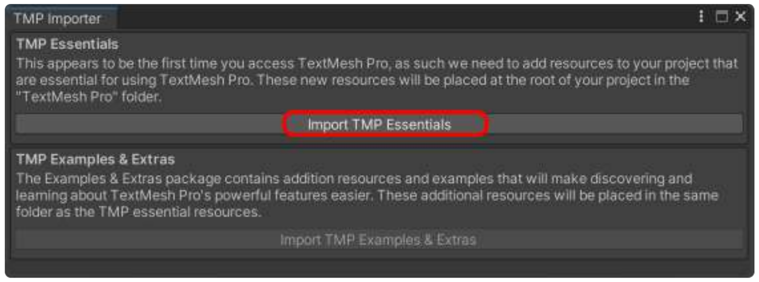
- ✓ Convai
  - ✓ ConvaiAR
  - ✓ Scenes
  - ✓ Scripts
  - ✓ Settings
- ✓ XR

4. Select all settings to be changed in the Project Settings and complete the installation by clicking " **Import** "



- ✓ ProjectSettings
  - ✓ EditorBuildSettings.asset
  - ✓ GraphicsSettings.asset
  - InputManager.asset
  - ✓ ProjectSettings.asset
  - ✓ QualitySettings.asset

5. Open the " *Convai / Scenes / Convai Demo - AR* " demo scene. If the TMP Importer window appears (It will appear if TMP Essentials is not installed in your project), click " **Import TMP Essentials** " to install TextMeshPro Essentials for UI text objects.



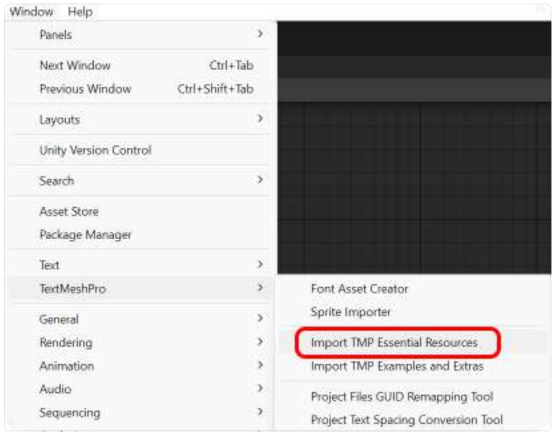
TMP Essentials

This appears to be the first time you access TextMesh Pro, as such we need to add resources to your project that are essential for using TextMesh Pro. These new resources will be placed at the root of your project in the "TextMesh Pro" folder.

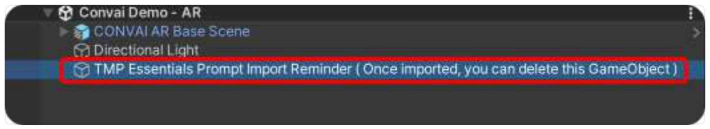
TMP Examples & Extras

The Examples & Extras package contains addition resources and examples that will make discovering and learning about TextMesh Pro's powerful features easier. These additional resources will be placed in the same folder as the TMP essential resources.

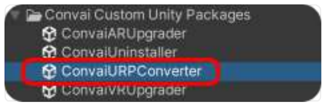
Alternatively, you can use the " *Window / TextMeshPro / Import TMP Essential Resources* " to install it.



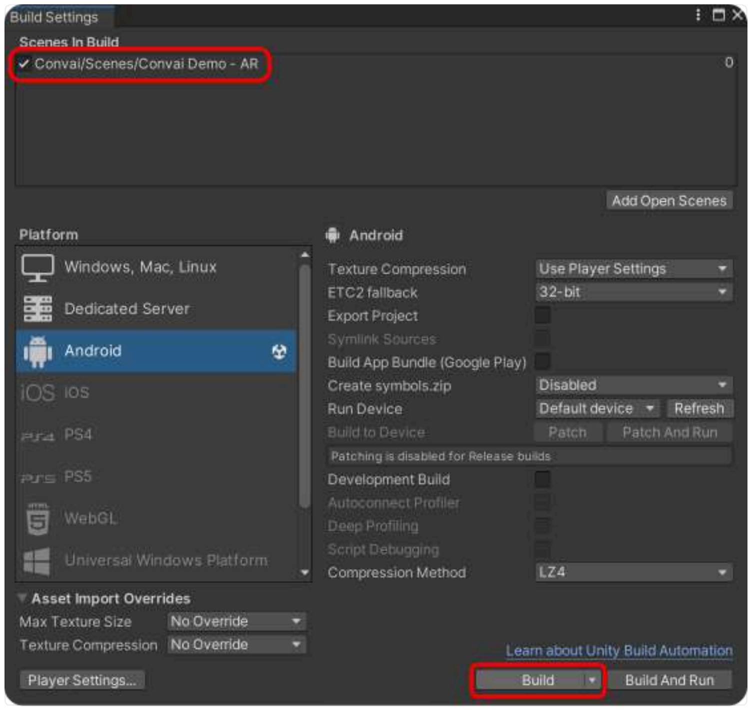
6. After importing TMP Essentials, you can remove the empty GameObject in your scene that triggers the Prompt window to appear.


Convai Demo - AR
CONVAI AR Base Scene
Directional Light
TMP Essentials Prompt Import Reminder (Once imported, you can delete this GameObject)

7. If you see 3D objects in pink, it's a shader issue. If you're using URP, convert the materials to URP by double-clicking on " *Convai / Convai Custom Unity Packages / ConvaiURPConverter* " and importing all assets in the window that appears.



8. Ensure you've set up your API Key (Convai / Convai Setup).
9. Build your project by going to " *File / Build Settings / Build* " Ensure that the " **Convai Demo - AR** " scene is included in the Scenes in **Build** section.



✓ Convai/Scenes/Convai Demo - AR 0

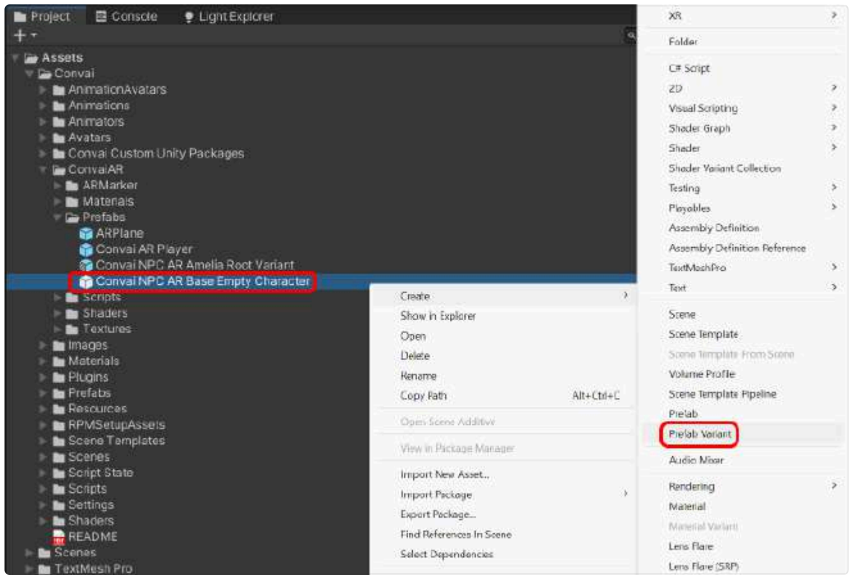
Texture Compression: Use Player Settings
ETC2 fallback: 32-bit
Create symbols.zip: Disabled
Run Device: Default device
Compression Method: LZ4
Max Texture Size: No Override
Texture Compression: No Override

Now everything is ready for testing. 🙂 ✅

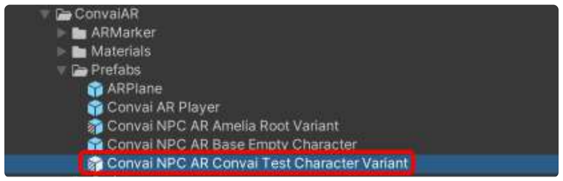
How to Add and Adjust Size Of My Own Character?

If you've created a Ready Player Me character on convai.com playground and want to add it to your AR project, follow these steps:

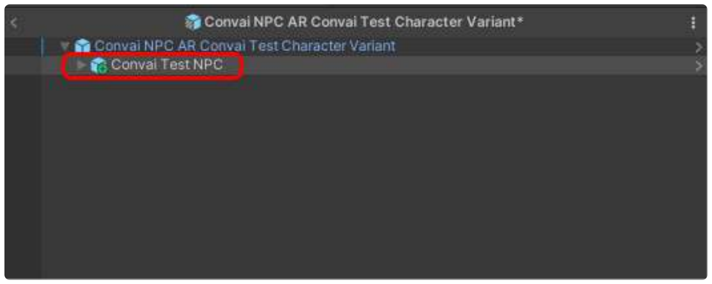
1. Right-click on the " *Convai / ConvaiAR / Prefabs / Convai NPC AR Base Empty Character* " prefab.
2. Click on " *Create / Prefab Variant* "



3. You'll see a prefab variant created for " **Convai NPC AR Base Empty Character** "
4. Double-click on this prefab variant.

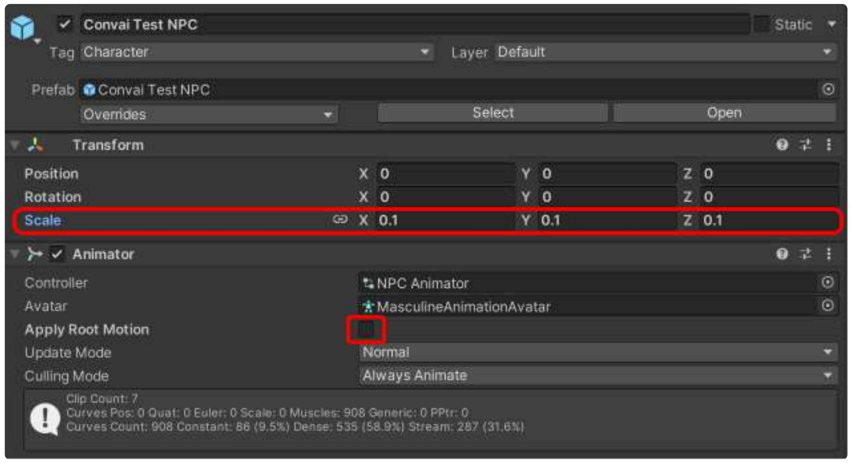

ConvaiAR
ARMarker
Materials
Prefabs
ARPlane
Convai AR Player
Convai NPC AR Amelia Root Variant
Convai NPC AR Base Empty Character
Convai NPC AR Convai Test Character Variant

5. In the Hierarchy section, add your imported character as a child to this prefab variant.


Convai NPC AR Convai Test Character Variant*
Convai NPC AR Convai Test Character Variant
Convai Test NPC

ⓘ Use the " Importing RPM Characters " guide to add your character to your project.

6. After adding your character, click on your character.
7. In the Inspector, adjust the Scale settings as needed. To prevent your character from moving with animation while talking, disable the " **Apply Root Motion** " option in the Animator.

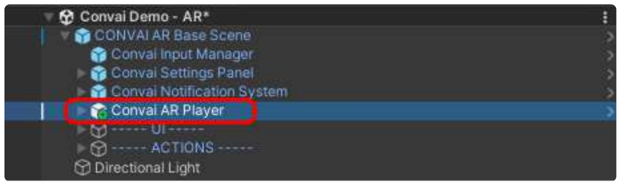

Convai Test NPC Static
Tag Character Layer Default
Prefab Convai Test NPC
Overrides Select Open

Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 0.1 Y 0.1 Z 0.1

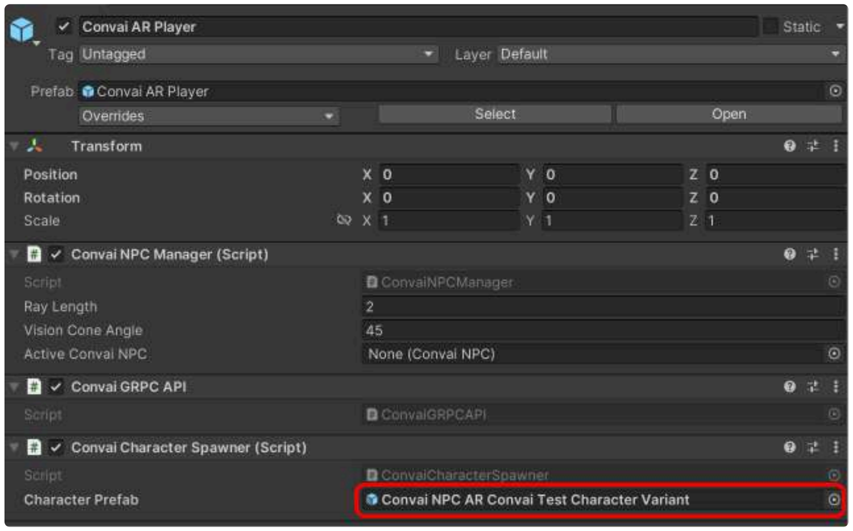
Animator
Controller NPC Animator
Avatar MasculineAnimationAvatar
Apply Root Motion ☐
Update Mode Normal
Culling Mode Always Animate

Clip Count: 7
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 908 Generic: 0 PPtr: 0
Curves Count: 908 Constant: 86 (9.5%) Dense: 535 (58.9%) Stream: 287 (31.6%)

8. After these steps, save your prefab variant by pressing **CTRL + S**.
9. Open the " *Convai / Scenes / Convai Demo - AR* " scene.
10. Click on the " **Convai AR Player** " object under " **ConvaiAR Base Scene** "



11. In the Inspector, under the " **Convai Character Spawner** " component, add your prefab variant to the **Character Prefab** field.


Convai AR Player
Tag Untagged Layer Default
Prefab Convai AR Player
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Convai NPC Manager (Script)
Script ConvaiNPCManager
Ray Length 2
Vision Cone Angle 45
Active Convai NPC None (Convai NPC)
Convai GRPC API
Script ConvaiGRPCAPI
Convai Character Spawner (Script)
Script ConvaiCharacterSpawner
Character Prefab Convai NPC AR Convai Test Character Variant

Now, everything is ready to test your character in the AR environment!🙂✅

Creating this prefab variant is to prevent automatic scaling (1,1,1) of your prefab when instantiated in the AR environment.

To avoid issues with scale adjustments, we added our character as a child to an empty parent object. For convenience, we created an empty prefab variant.

Building for macOS Universal apps

Overview

When building Unity projects for macOS, developers may encounter issues with microphone permissions, particularly when targeting both Intel and Apple Silicon Macs. This document outlines the problem, symptoms, causes, and solutions to help ensure successful access to the microphone across different Mac architectures.

Problem Description

Some users have reported that while building macOS universal apps, Apple Silicon Macs handle microphone permissions without issue, while Intel Macs may fail to access the microphone due to differences in architecture. This can result in a lack of microphone response, DLL not found Exceptions, error messages, potential application crashes, or no audio input being detected.

Cause

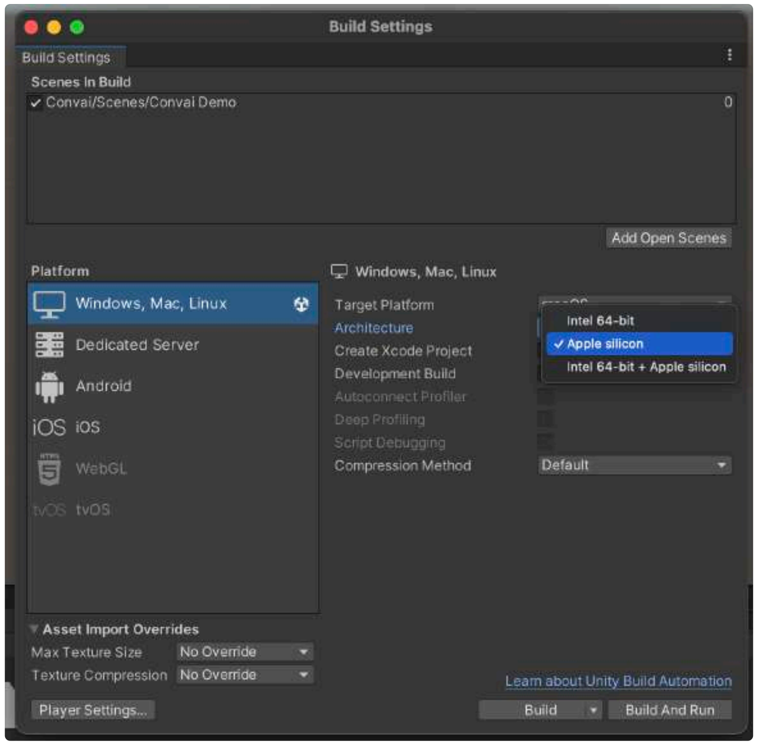
The issue stems from the `grpc_csharp_ext.bundle`, which is crucial for networking in Unity projects. There are separate versions of this DLL for Intel and Apple Silicon architectures, and they cannot be easily merged or applied universally. The grpc library currently lacks dedicated support for resolving these dll issues in Unity.

Solutions

Standalone Build Settings

- **For Intel Macs:** Use Standalone builds targeted specifically for the Intel architecture to ensure compatibility.

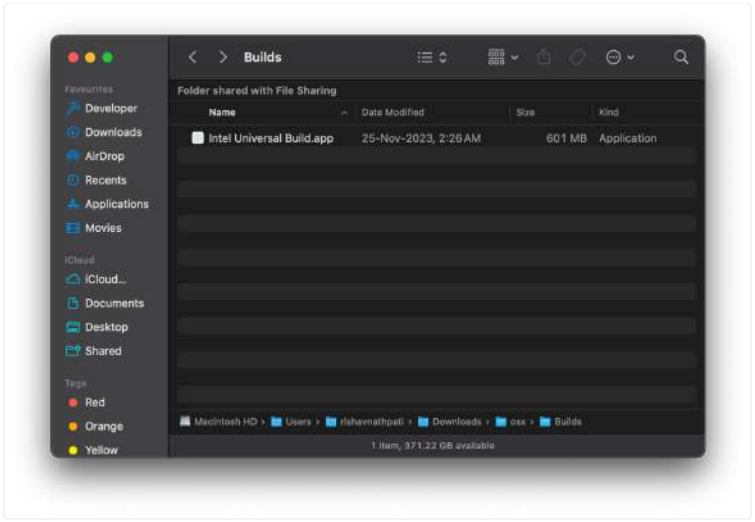
- **For Apple Silicon Macs:** Prefer Standalone builds for the ARM64 framework for optimal performance, although Universal builds are also an option.



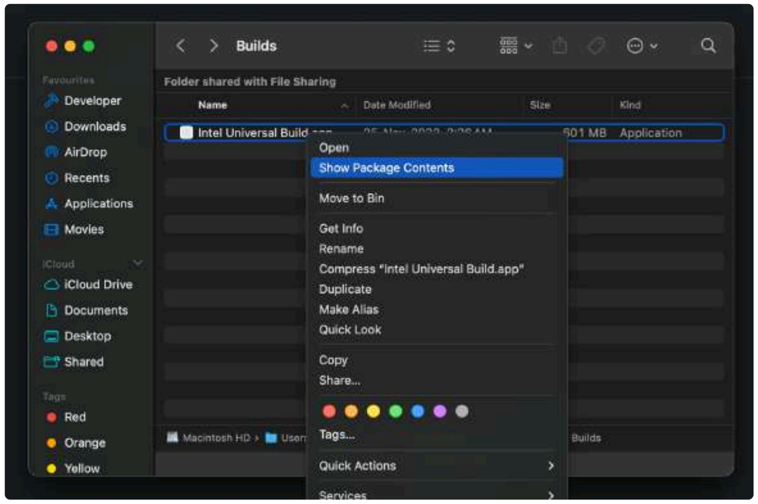
New Procedure for Universal Builds from Intel Macs

After completing a Universal build on an Intel Mac, you must manually update the `grpc_csharp_ext.bundle` to ensure proper functionality. Follow these steps:

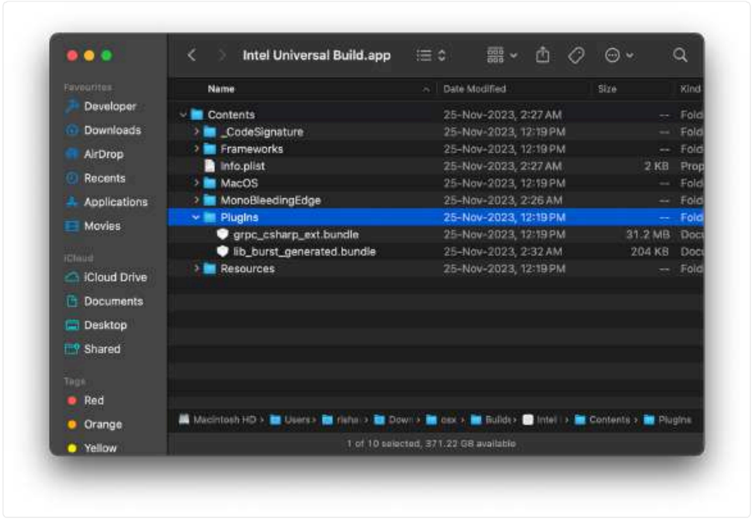
1. Locate the `.app` file generated by the build process.


Builds
Folder shared with File Sharing
Name
Date Modified
Size
Kind
Intel Universal Build.app 25-Nov-2023, 2:26 AM 601 MB Application
Macintosh HD › Users › rishavnathpati › Downloads › osx › Builds
1 item, 371.22 GB available

2. Right-click on the `.app` file and select "Show Package Contents."

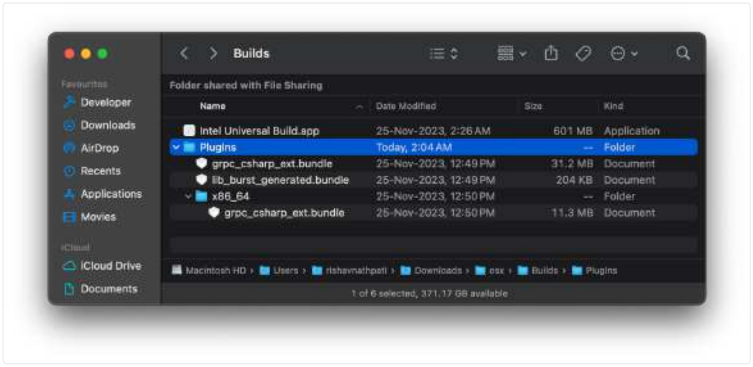


3. Navigate to the `Contents/Plugins` folder within the package.

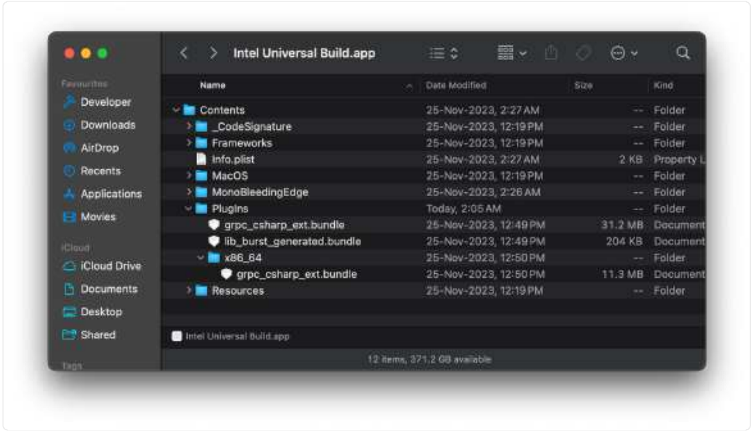


```
Name                          Date Modified            Size   Kind
Contents                      25-Nov-2023, 2:27 AM     --     Fold
  _CodeSignature              25-Nov-2023, 12:19 PM    --     Fold
  Frameworks                  25-Nov-2023, 12:19 PM    --     Fold
  Info.plist                  25-Nov-2023, 2:27 AM     2 KB   Prop
  MacOS                       25-Nov-2023, 12:19 PM    --     Fold
  MonoBleedingEdge            25-Nov-2023, 2:26 AM     --     Fold
  PlugIns                     25-Nov-2023, 12:19 PM    --     Fold
    grpc_csharp_ext.bundle    25-Nov-2023, 12:19 PM    31.2 MB Doc
    lib_burst_generated.bundle 25-Nov-2023, 2:32 AM    204 KB Doc
  Resources                   25-Nov-2023, 12:19 PM    --     Fold

Macintosh HD › Users › risha › Down › osx › Builds › Intel › Contents › Plugins
1 of 10 selected, 371.22 GB available
```

4. Replace the contents of this folder with the components from the provided plugin folder (link to plugin folder).



```
Folder shared with File Sharing
Name                          Date Modified            Size    Kind
Intel Universal Build.app     25-Nov-2023, 2:26 AM     601 MB  Application
Plugins                       Today, 2:04 AM           --      Folder
  grpc_csharp_ext.bundle      25-Nov-2023, 12:49 PM    31.2 MB Document
  lib_burst_generated.bundle  25-Nov-2023, 12:49 PM    204 KB  Document
  x86_64                      25-Nov-2023, 12:50 PM    --      Folder
    grpc_csharp_ext.bundle    25-Nov-2023, 12:50 PM    11.3 MB Document

Macintosh HD › Users › rishavnathpati › Downloads › osx › Builds › Plugins
1 of 6 selected, 371.17 GB available
```



```
Name                          Date Modified            Size    Kind
Contents                      25-Nov-2023, 2:27 AM     --      Folder
  _CodeSignature              25-Nov-2023, 12:19 PM    --      Folder
  Frameworks                  25-Nov-2023, 12:19 PM    --      Folder
  Info.plist                  25-Nov-2023, 2:27 AM     2 KB    Property L
  MacOS                       25-Nov-2023, 12:19 PM    --      Folder
  MonoBleedingEdge            25-Nov-2023, 2:26 AM     --      Folder
  Plugins                     Today, 2:05 AM           --      Folder
    grpc_csharp_ext.bundle    25-Nov-2023, 12:49 PM    31.2 MB Document
    lib_burst_generated.bundle 25-Nov-2023, 12:49 PM   204 KB  Document
    x86_64                    25-Nov-2023, 12:50 PM    --      Folder
      grpc_csharp_ext.bundle  25-Nov-2023, 12:50 PM    11.3 MB Document
  Resources                   25-Nov-2023, 12:19 PM    --      Folder

Intel Universal Build.app
12 items, 371.2 GB available
```

Important: The `grpc_csharp_ext.bundle` may not be included correctly in the final build when built from an Intel Mac. Always verify that the `Plugins` folder in the build contains the correct DLLs. If there is any confusion or the DLLs are missing, replace or add the contents of the `Plugins` folder with the one provided by us.

Conclusion

Building for macOS requires careful consideration of the distinct Intel and Apple Silicon architectures. The current best practice is to use Standalone build settings tailored to the specific architecture of the target Mac. As we progress, we will have a more integrated solution for managing DLLs that will simplify the development process for universal macOS applications.

Changelogs

Convai Unity Plugin Changelogs - Stay updated with the latest changes.

Version 3.0.1 current

Released: Jun 21, 2024

Bug Fixes:

- Fix macOS TMP UGUI render issue in demo scene
- Prefab missing animator
- Updated ActiveNPC layer check logic

Version 3.0.0

Released: Jun 13, 2024

What's Changed

NPC2NPC System

- Implemented NPC2NPC conversation flow system.
- Added handling for conversation interruptions and restarts.
- Enhanced conversation history tracking and flow management.

Narrative Design

1. Added Narrative Design-related files and trigger narrative section function.
2. Refactored Narrative Design API, created new behavior trees for movement and added section change events.

3. **Folder Restructuring**
   - Complete folder and scripts folder restructure
4. **Gender-Based Animator:** Added gender-based Animator controller
5. **Feedback System**
   - Implemented a feedback system with thumb icons and animations
   - Updated Transcript UI Prefabs with feedback buttons
6. **Convai Custom Packages:** Updated Convai Custom Package Installer and added iOS DLL Downloader
7. **Scene Perception:** Added feature to allow players to point at game objects and talk about them

Improvements

1. **Texture and Material Compression**
   - Compressed Amelia and other images (POT and Crunch)
   - Updated image names and removed unused image assets
2. **UI Updates**
   - Updated UI prefabs, including Transcript UI, Mobile UI, and Mobile QA UI
   - Transcript UIs and text updated
   - Updated logos and logo paths
3. **System Improvements**
   - Refactored Lipsync system with added teeth support and implemented facial expression proto files
   - Updated ConvaiURPConverterPackage, burst and TMPro packages; Convai Custom Package Installer/Exporter
   - Updated NavMesh and NPC2NPC character rotation
   - Added new demo scene and RPM characters
   - Updated various demo scenes for consistency
4. **Microphone Manager:** Updated Microphone Manager to a singleton class
5. **API Key Access:** Simplified API Key access

6. **Convai Scene Template:** Created new scene template and dynamic input system assigner
7. **Demo Scenes**
   - Added NPC2NPC demo scene
   - Added Narrative Design demo scene
   - Added new demo scene with all features encompassed
   - "Convai Essentials" prefab for desktop and mobile
8. **Lipsync Overhaul**
   - Overhauled lipsync system, added AR-Kit and Reallusion character support
   - Updated version and added various improvements to frame processing
9. **Input System**
   - Added new input system pragma checks and Convai Character Layer
   - Simplified Input Manager and ensured future-proofing

Bug Fixes

1. **Transcript UI Bug Fixes:** Fixed bugs and improved system for Transcript UI character list
2. **Microphone Permission:** Fixed Android and iOS microphone permission issues

Version 2.1.0

What's Changed

1. VR Support: Implement Virtual Reality features to create a fully immersive experience with the press of a button.
2. AR Support: Integrate Augmented Reality capabilities, allowing characters and environments to interact with you in the real world with the press of a button.
3. Settings Panel: Introduce a comprehensive settings panel that allows users to customize their experience.

4. Microphone Test System: Incorporate a microphone testing feature to ensure optimal audio input quality.
5. Notification System: Implement a robust notification system to inform users of in-game events - specifically microphone-based issues.
6. Input Manager: Develop a custom input management system that supports various input devices such as keyboards, gamepads, and touchscreens using Unity's new Input System.

Bugs and Improvement

1. Fixed: Head Tracking Doesn’t Work Without Action Component issue fixed.
2. Improvement: Added support for a customizable and dynamic Chatbox.
3. Improvement: Improved Lip-Sync Smoothing and audio-visual synchronization.
4. Improvement: Implement Action Events and Event Callbacks.
5. Improvement: Improved Logging System.
6. Improvement: Added ability to interrupt Character Response with Voice Interruption.
7. Improvement: Improved mobile platform transcription UI.

Version 2.0.0

What's Changed

1. Lip-sync: Integrate off-the-shelf Lip-sync for Reallusion and Oculus-based Characters.
2. Text-in Voice-out: Chat with the character using text.
3. Character Importer: Import Ready Player Me characters created on the Convai Playground.
4. Feature Control System: Enable Convai features as needed through the Convai NPC component.
5. Logging System: Have better control over what Convai information you see on the debug console.

6. Enhanced player controller: Automatically triggers the characters when you focus on them and then deactivates them when your focus has shifted.
7. URP Upgrader: Upgrade the Render Pipeline to Universal Render Pipeline with the URP Upgrader package (present in the Convai Folder).
8. UI Improvements: Improved user experience with automatically fading UI canvas.

Bugs and Improvement

1. Fixed: Unlocking the cursor will still cause the first-person camera to move around.
2. Fixed: Exiting play mode before the character is done speaking will cause Unity to crash or not complete compilation.
3. Fixed: Extra space between multiple chunks of text in the UI Text Fields.
4. Fixed: Actions crashing the Android scene.
5. Fixed: Empty responses from the server will not crash the game but only throw an error.
6. Improvement: Smoothened Blinking.
7. Improvement: Smoothened Gaze-Following-based Neck movement.
8. Improvement: Plugin structure reorganization.