

COMSM0127: Immersive Interaction and Audio Design

Lab 8: Intelligent Virtual Assistant

The objective of these labs is to make you aware of current communication technologies for interacting with both people and AI. The labs consist of two parts: i) we will guide you on implementing text-based Intelligent Virtual Assistants (IVA) that can be further used with avatars to create more realistic interactions, ii) we will provide an example of how to use this IVA tool to generate code in Unity. This example will give you a better understanding of how to utilise the IVA for advanced applications. However, please note that the model is primarily designed for language formatting, and not for generating code directly. As a result, you may encounter some errors that require you to review and adjust the generated code.

Guided Instruction

Task 1: Chatbot

Here we develop text-based IVA using [ChatGPT](#), which is a large language model (LLM) developed by OpenAI. Specifically, we will be utilising [ChatGPT](#) to request access through its API and to answer questions.

1. Create an OpenAI account ([Link](#)) if you don't have one.
2. Adapting from the model provided by Dilmer, we provide you with the modified source code to be download from [HERE](#).
3. Open the downloaded project in Unity Hub (ignore Safemode).
4. Delete the default 'Untitled' Scene and drag Assets > Scenes > ChatGPT to the Hierarchy.
5. Open the script from Assets > Scripts > ChatGPT > ChatGTPSettings.cs, add strings for **apiModel** and **apiOrganization**.
6. **Complete the ChatGPTResponse.cs script.** You should try to write a **ChatGPTResponse.cs** script yourself according to the response from ([ChatGPT API reference](#)) to deserialize the JSON data into C# objects.

Hint: Ask ChatGPT to write a ChatGPTResponse script by providing the response of API reference. A C# representation based on this JSON object (Copy the Response code on the page) including JSON property attribute names.

7. **Check the ChatGPTRequest.cs script.** Check the propertyName of JsonProperty includes the "model" and "messages". And make sure that the object of ChatGPTMessage

connects to the ChatGPTResponse > ChatGPTMessage. This will ensure that we can communicate with ChatGPT.

8. Complete the ChatGPTClient script.

- a) Remove the prompt line and add the script as follow. This will take a request which includes a model and an array of the messages.

```
Model = chatGTPSettings.apiModel ,
Messages = new ChatGPTMessage[]
{
    new ChatGPTMessage
    {
        Role = "user",
        Content = prompt
    }
}
```

- b) Add the following two lines to 'SetRequestHeader' for authenticating against an OpenAI API using an API key and organization.

```
request.SetRequestHeader("Authorization",
$"Bearer chatGTPSettings.apiKey");
request.SetRequestHeader("OpenAI-Organization",
chatGTPSettings.apiOrganization);
```

9. Select the ChatGPTClient GameObject, add the Assets > Settings > ChatGPT > ChatGTPSettings.asset to Chat GPT Settings in the inspector.
10. Set the **Chat GPT Settings** for **ChatGPTClient**.
 - a) **Open the OpenAI website to get the API key.** Generate a new API key by pressing the button create new secret key, and remember to copy the API keys (only appear once) to the ChatGTPSettings > API key.
 - b) **Copy the Organization ID.** Click Settings on the left to find and copy the Organization ID to ChatGTPSettings > API Organization.
 - c) Copy the Link <https://api.openai.com/v1/chat/completions> to ChatGTPSettings > API URL, and set the ChatGTPSettings > API model for gpt-3.5-turbo.

11. Select the ChatGPTTester GameObject (not the script), you can edit the prompt Prefix Constant, prompt and reminders for the request of ChatGPT.

There are some example requests you can choose from the Chat GPT Question in the ChatGPTTester.

- why VR is the future?
- Speak to me "Hello World"

12. Now enter the play mode and try to talk with Chat GPT!

Hint: You might want to add 'In short' or 'Summarise' as ChatGPT is quite wordy (expensive).

Task 2: AI Generator

Now we ask ChatGPT to generate Unity C# script and compile the generated code using Roslyn compiler in Play mode.

1. Open Assets > Scripts > ChatGPSTester.cs, and uncomment the last five lines (Process And Compile Response) for compiling the C# Code generated by the ChatGPT.
2. Select the GameObject ChatGPTLogger > CompileButton. In the Button > On Click(), Select the + button to add a new action.
3. Click and drag the GameObject ChatGPSTester from the Hierarchy onto the empty Object field to assign it.
4. Change the No Function to ChatGPSTester.ProcessAndCompileResponse
5. Select the ChatGPSTester GameObject(not the script), you can edit the prompt Prefix Constant, prompt and reminders for the request of ChatGPT.

There are some example requests you can choose from the Chat GPT Question in the ChatGPSTester.

- Print "Hello world"
- Generate rotating cubes
- Loads a PlayerArmature from Resources consistently every 1/2 second, get StarterAssetsInputs component, set the "move" field to a vector2 with 1.0f for x and 0 for y, then set the "jump" field to true

6. Now enter the Play mode ask ChatGPT to generate the code and compile the code.

Finally, try to integrate this Panel into your VR room.

Task 3 [Stretch Goal]: Voice Chat

This task we use Photon Voice 2 ([Link to document](#)) as example. Photon Voice 2 Photo Voice 2 ([Link](#)) is a real-time multiplayer voice communication using cloud-based system. Here is an example of implementing Photo Voice on Quest.

GitHub: [Link](#)

How to:

- Part 1: [Link](#)
- Part 2: [Link](#)
- Part 3: [Link](#)
- Part 4: [Link](#)

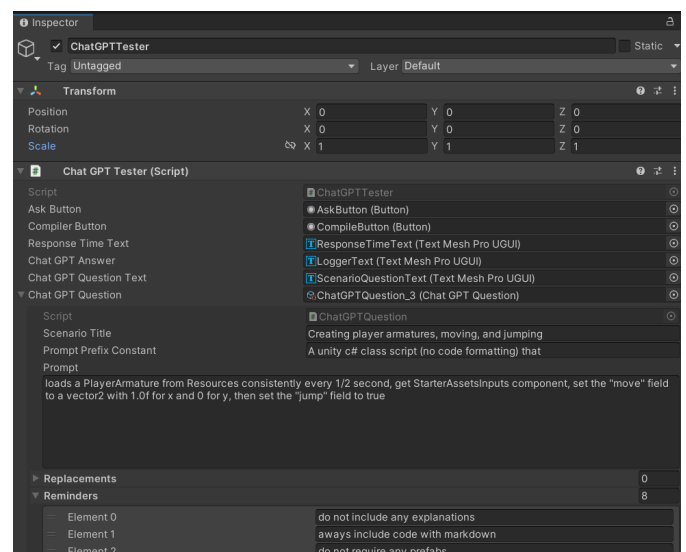


Figure 1: A request prompt on the script of Inspector