Project Title:
AI-Powered Interactive Poetry Generation Application

Project Description:
This project aims to create an interactive Unity application where users can engage with a virtual character representing the CEO of SDAIA, **Abdullah Al-Ghamdi**, who can generate and recite poetry in the style of three renowned Arab poets: **Badr bin Abdulmohsen**, **Khaled Al-Faisal**, and **Khlaf bin Hathal**. Each poet has their **unique recitation style** and a distinct virtual avatar that reflects their character.

Users can also **ask questions** to the virtual character, which responds using AI-based answers. The application leverages IBM Watson and OpenAI for text generation and voice processing, creating a rich, interactive experience.

Project Components:

1. **Virtual Character (Avatar):**
    - The avatars are visually designed in Unity to reflect each poet's unique persona and spirit.
2. **AI Poetry Generation:**
    - Utilizing **OpenAI's model** to train AI on the distinctive styles of each poet by feeding it a collection of their poems.
    - The model generates new poetic lines based on user prompts, allowing customization by theme (e.g., love, nostalgia, wisdom).
3. **Voice Technology:**
    - **IBM Watson Text-to-Speech** is used to convert generated text into audio, reciting poems in a voice that matches each poet's style.
    - Additional Speech-to-Text feature for converting user speech into text, allowing voice-based questions to the character.
4. **Interactive Question and Answer System:**
    - Using **IBM Watson Assistant** to process user questions (text and voice) and generate interactive responses based on AI-powered data.
5. **User Interface and Experience (UI/UX):**
    - Designing a flexible, appealing interface that lets users select a poet, ask questions, and listen to poetry.
    - Adding customization options such as adjusting the speed and tone of the voice for a more realistic, user-friendly experience.

Use your own avatar

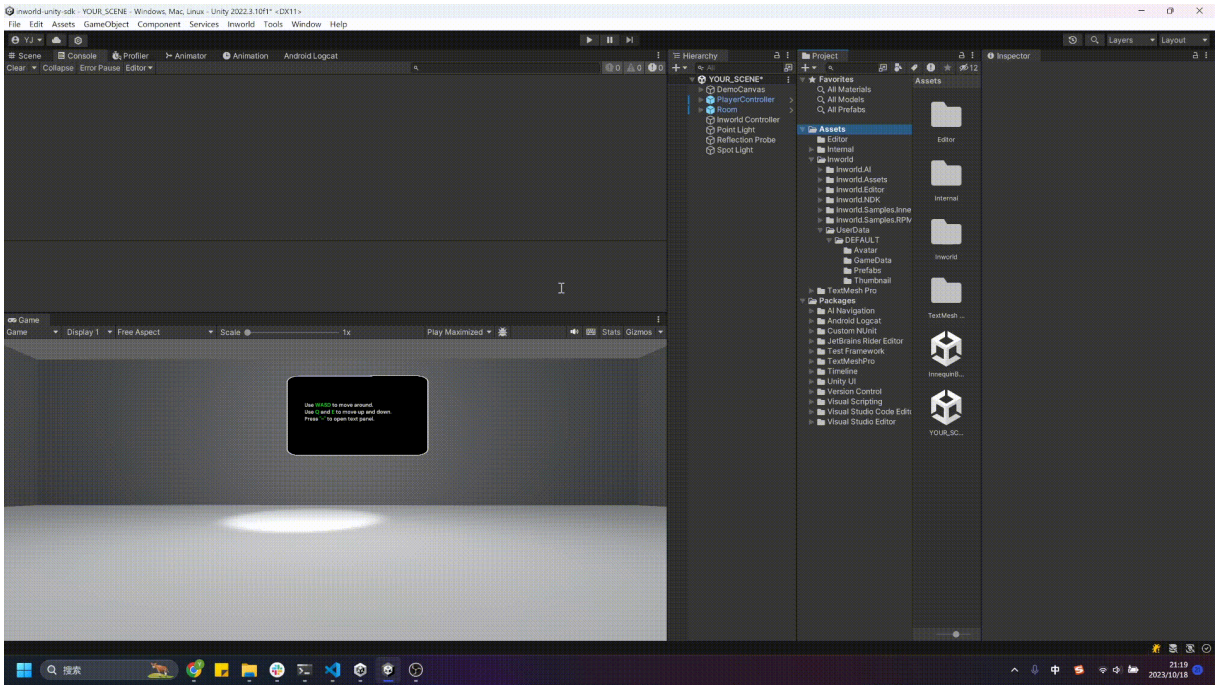
In Inworld Unity SDK 2.0.3 or later, we have made it much easier to integrate your own avatars with our characters. Let's take a look.

1. Drag your character prefab into the scene

Once you've finished downloading an Inworld scene through the **Inworld Studio Panel**, you can use these characters across any Unity scene without having to return to the panel.

You can go to the `UserData > DEFAULT > Prefabs` folder to find your character prefabs, from there you can drag them into your Unity scene. Ensure that the **InworldController** in your Unity scene contains the correct **Game Data** for your characters.

⚠️ **Note**: The **InworldController** will be automatically created if not already initialized, and will contain the current **Inworld Scene** you've set in **Inworld Studio Panel**.



2. Convert any object to an Inworld Character

If you want to use another GameObject as the model for your character, you can do the following:

1. Create an existing Inworld Avatar.

2. Add your object as a child to the Inworld Character GameObject.

Create or drag the GameObject you wish to use as your character model under the Inworld Character in the hierarchy.

3. Disable original Armature.

Disable the existing `Armature` GameObject.

4. Done.

3. Use your own GLB humanoid avatar

If you have a GLB humanoid avatar, you can integrate with Inworld animations and lip syncing.

1. Create an existing Inworld Avatar.

2. Add your avatar as a child to the Inworld Character GameObject.

Drag your own avatar under the Inworld Character in the hierarchy.

3. Replace original Armature.

Rename your avatar to `Armature`, and delete or disable the existing `Armature`.

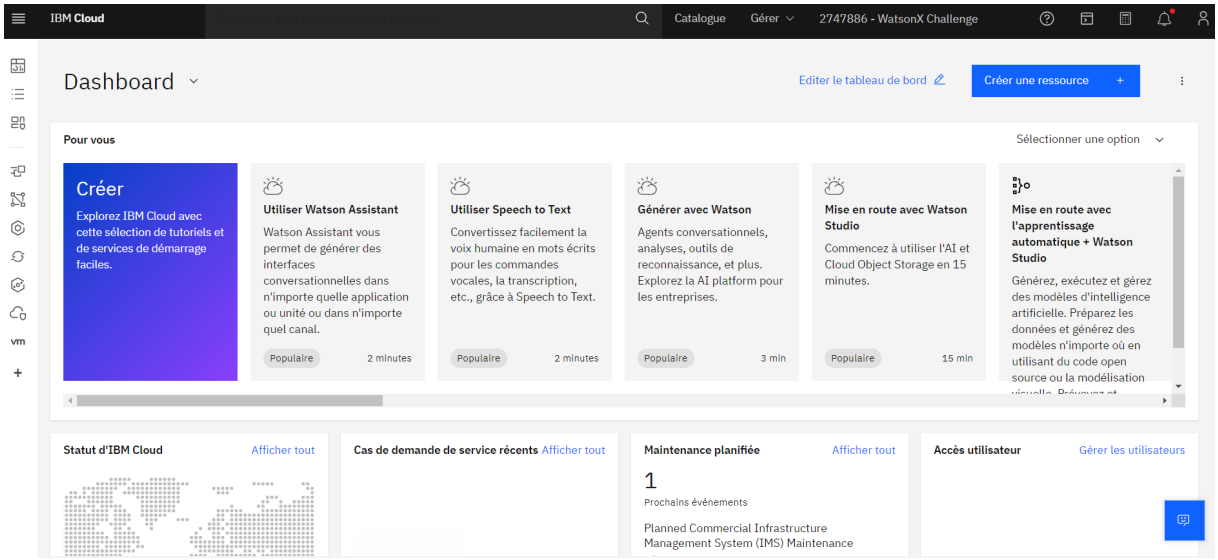
4. Done.

Save your current scene and press `Play` to see the result.

Custom voice Integration

In version 2.1.5 and higher, integrating custom voices has become much easier

Integrating AI with IBM Watson



Watson Setup:

- Create an IBM Cloud account.
- Activate Watson services such as Speech to Text, Text to Speech, or Conversation Assistant.
- Retrieve the API keys for each of the services.

IBM Watson SDK for Unity:

- Install the SDK in your Unity project and connect it with the API keys.

5. AI Features to Integrate

- **Speech-to-Text:** Convert the player's voice into text.
- **Text-to-Speech:** Make your avatar speak.
- **Conversation AI (Chatbot):** Allow the avatar to answer questions using IBM Watson Assistant.

6. Example Scripts Here's a simple example to get started with Text-to-Speech using IBM Watson:

```
csharp
Copier le code
using IBM.Watson.TextToSpeech.V1;
using IBM.Cloud.SDK.Authentication.Iam;

public class WatsonTextToSpeech : MonoBehaviour
{
    private TextToSpeechService _service;

    void Start()
    {
        IamAuthenticator authenticator = new IamAuthenticator(apikey:
"your-api-key");
        _service = new TextToSpeechService(authenticator);
        _service.SetServiceUrl("your-service-url");
        _service.Synthesize(OnSynthesize, "Hello, I am a smart avatar",
voice: "fr-FR_ReneeV3Voice", accept: "audio/wav");
    }

    private void OnSynthesize(DetailedResponse<byte[]> response, IBMError
error)
    {
        if (error == null)
        {
            AudioClip clip = WavUtility.ToAudioClip(response.Result);
            AudioSource audioSource =
gameObject.AddComponent<AudioSource>();
            audioSource.clip = clip;
            audioSource.Play();
        }
        else
        {
            Debug.LogError(error.ToString());
        }
    }
}
```

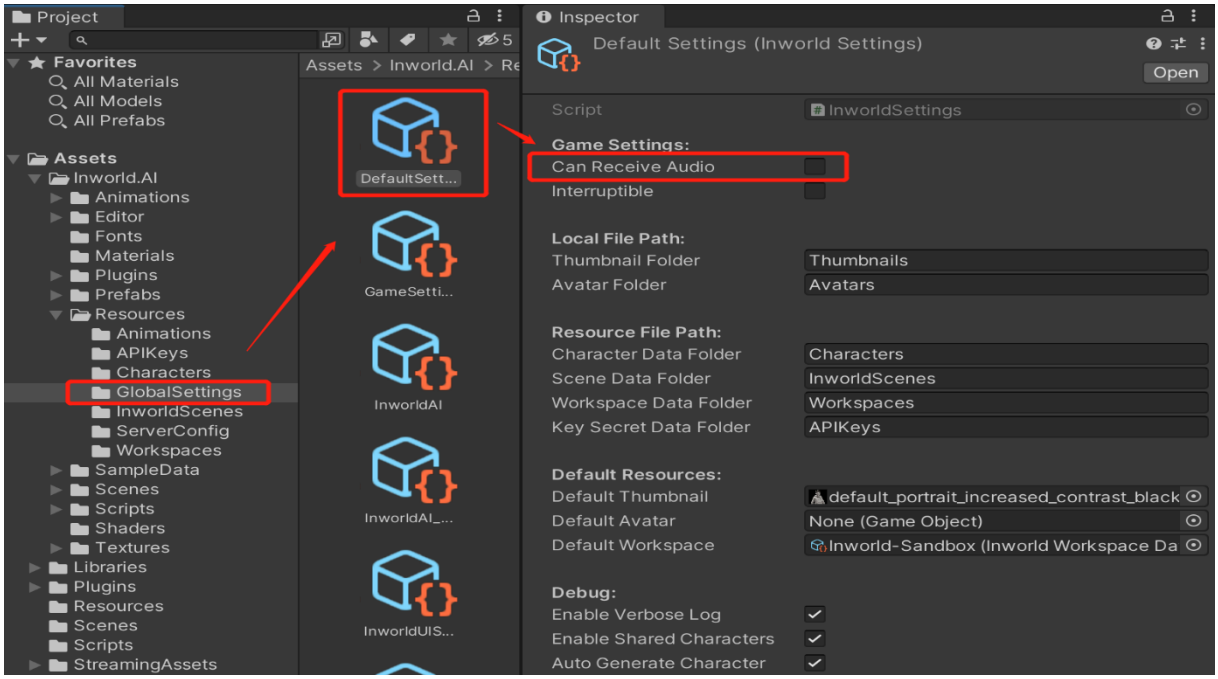
Text-to-Speech: This script converts a phrase into audio with Watson and plays the sound in Unity.

In this tutorial, we will guide you through the process of integrating your own text-to-speech (TTS) device.

1. Disabling Audio Receiving

Starting from version 3.0.0, you can right click in Project tab, then select `Inworld > Default Settings`, and turn off audio by unchecking `Audio` option.

If you are using previous version, after version 2.1.6, you can turn off audio by unchecking the `Can Receive Audio` option in `Default Settings`. See the screenshot below for more details.



For users on version 2.1.5 or lower, update `InworldSettings.cs` by simply setting the `Audio` of `Capabilities` to false.

```
/// <summary>
///     Returns the capabilities settings for communicating with Inworld Server.
/// </summary>
1 usage
public CapabilitiesRequest Capabilities => new CapabilitiesRequest
{
    Animations = true,
    Audio = false,
    Emotions = true,
    Gestures = true,
    Interruptions = true,
    Text = true,
    Triggers = true,
    TurnBasedStt = true,
    PhonemeInfo = true
};
```

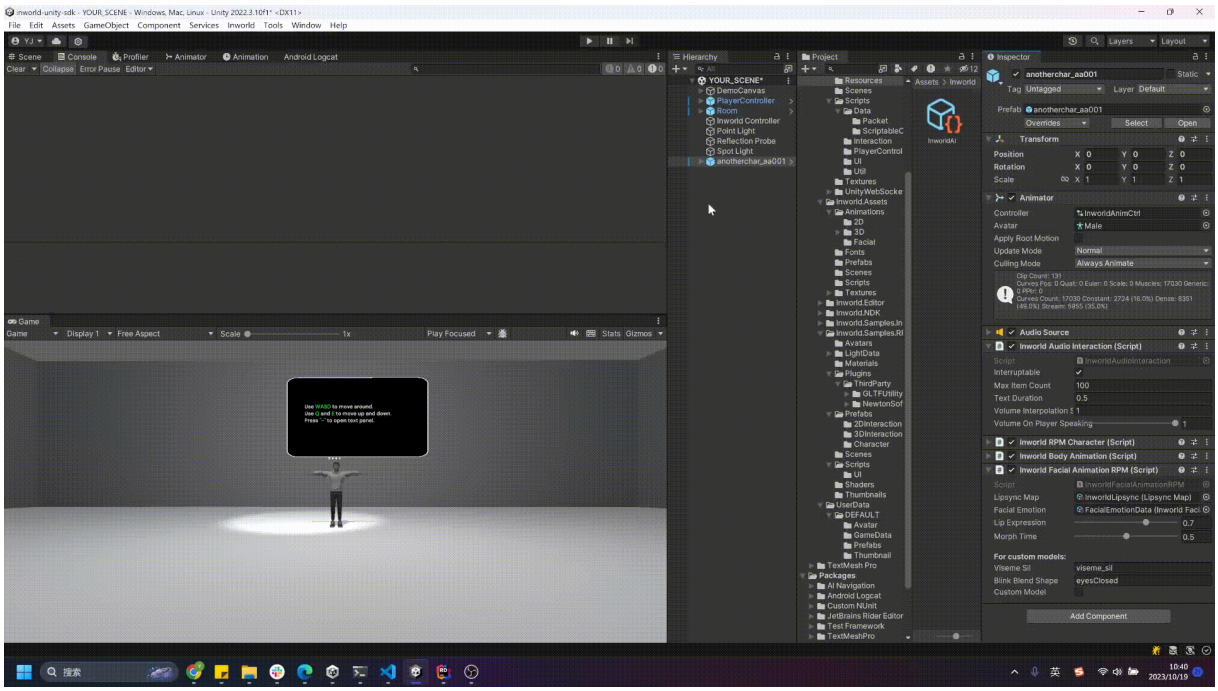
Note: This will prevent audio from Inworld's TTS interacting with your project, after applying this setting, the server will no longer send audio.

2. Replacing Interactions on the Character

Since we have disabled audio, our character's default **InworldAudioInteraction** will not work, because it requires `AudioClips` to generate bubbles.

We need to replace **InworldAudioInteraction** with **InworldInteraction**, which is text-based. Here are the following steps:

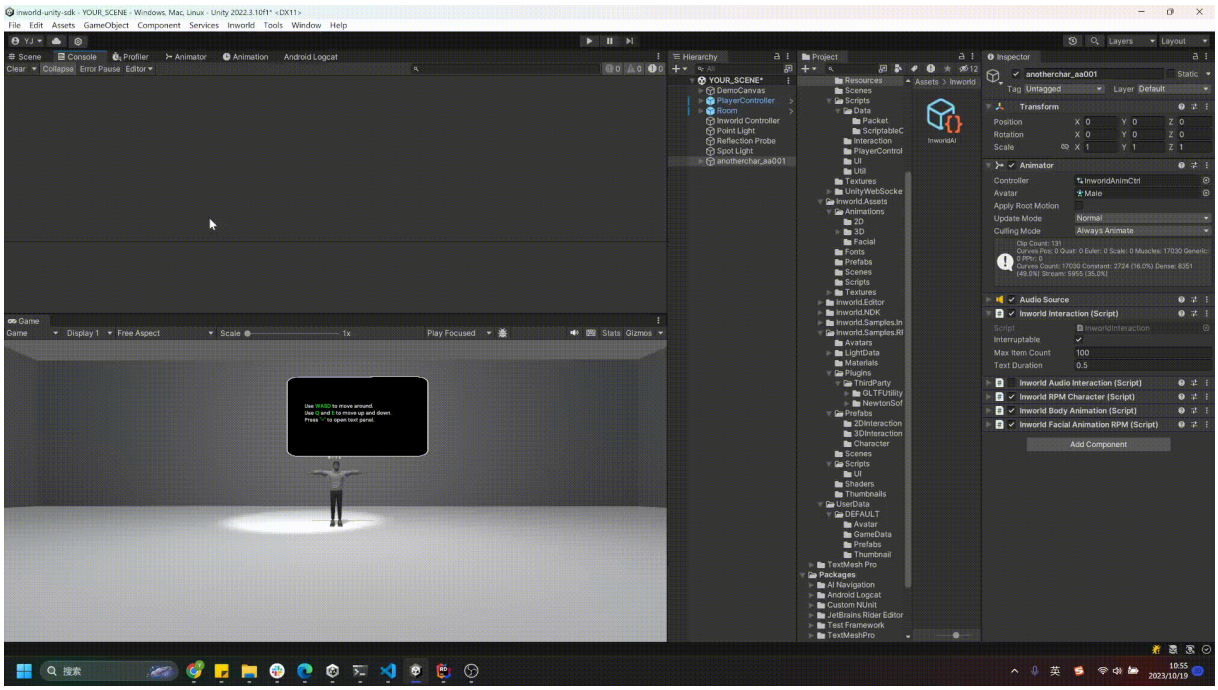
- i. Completely unpack the character prefab.
- i. Add the **InworldInteraction** component and place it at the top of the component stack, above the **InworldAudioInteraction**.
- i. Disable the **InworldAudioInteraction** component.



3. (Optional) Enable Log of Utterance

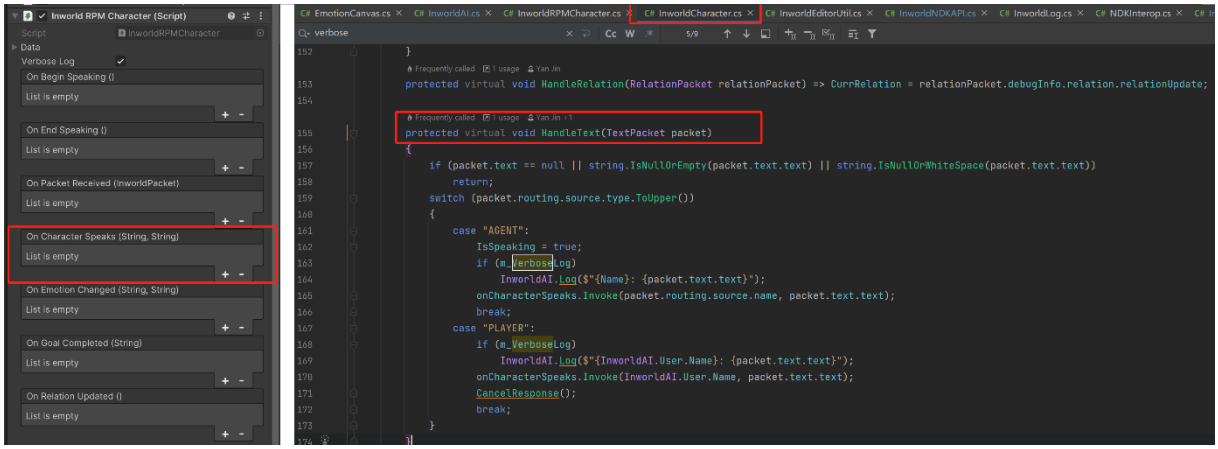
You can toggle the `Verbose Log` option under the `InworldCharacter` script of the attached character.

After starting the game, the character's text will be displayed on the console panel, you can extend this to a subtitle system or tie it into event triggers.



4. Register your own TTS Service

You can either register the event on `InworldCharacter`'s `OnCharacterSpeak`, or overwrite the script `HandleText` to get the text directly.



Here is an example of how to use this repo: UnityRuntimeTextToSpeech's API:

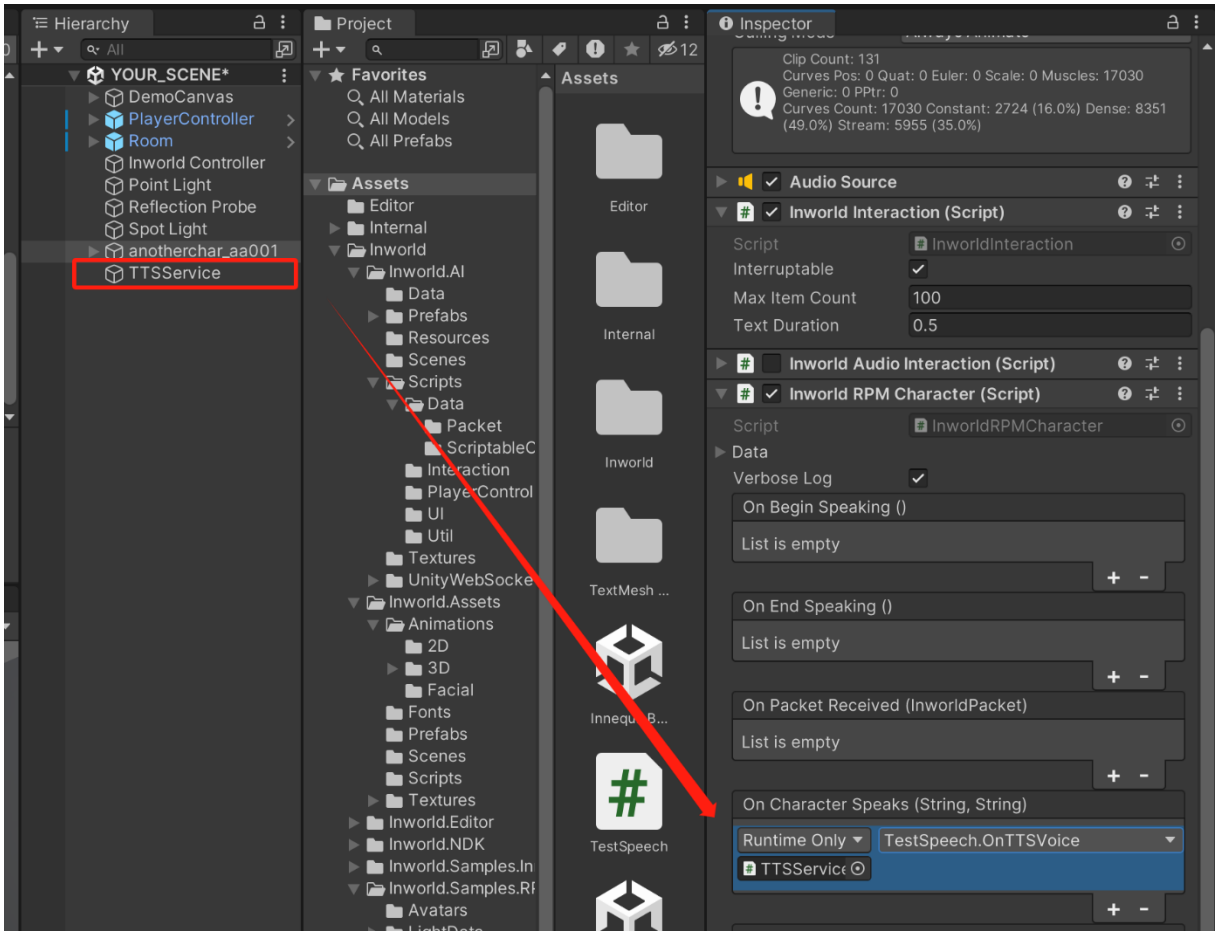
```
public class TestSpeech : MonoBehaviour
{
    public string sayAtStart = "Welcome!";

    // Start is called before the first frame update
    void Start()
    {
        // TEST speech
        Speech.instance.Say(sayAtStart, TTSCallback);

    }
    ...
```

```
```

According the API above, you need to create a public function with the following:

```
    public void OnTTSVoice(string character, string content)
    {
        if (character != "Player")
            Speech.instance.Say(content, TTSCallback); // Referring to your
own API
    }
```

⚠️ **Note:** In the `OnCharacterSpeaks` UnityEvent, the first parameter is the name of the speaking character, and the second parameter is the actual spoken content.
Then, add this function to `InworldCharacter`'s `OnCharacterSpeaks`



Exporting the Unity Project

Configure Export Settings:

- Go to **File > Build Settings** and configure the settings for exporting to **Windows**.
- Select **Windows** as the target platform, as Holobox typically uses a Windows system.
- Make sure to set the **resolution** and **graphics options** to take advantage of Holobox's capabilities (high resolution and multi-angle display).

Export the Project:

- Use **Build and Run** to generate the executable for Windows. This will create a **.exe file** that you will use for the Holobox.

3. Configure the Holobox

Install Necessary Drivers and Software:

- Ensure the Holobox has the required drivers and software to run the application (such as **DirectX** and **Unity Player**).
- The **Holobox Windows 86 inch** may require specific configurations for the holographic display, so refer to the Holobox documentation for technical details.

4. Transfer and Test the Avatar on Holobox

Transfer the Executable File:

- Connect to the Holobox via **USB** or network and transfer the **.exe file** you generated.

Test the Application:

- Run the file on the Holobox and ensure that the avatar displays correctly in 3D, considering the angle and depth of the holographic display.

AI Interaction:

- If the avatar uses AI, verify that cloud services (such as **IBM Watson**, **Inworld AI**, etc.) are working correctly on the Holobox.

5. Optimization

Performance and Graphics:

- The Holobox is a powerful machine, but **graphic optimization** is crucial to ensure the avatar runs smoothly.

Test Sensors (if available):

- If the Holobox has built-in sensors or cameras to interact with the audience, test them and make sure the avatar responds correctly.

6. Final Deployment

Once everything works well, you can set the Holobox to automatically launch the application every time it powers on.