Unity DeepSeek API Integration

Figure 1: DeepSeek Unity Integration

A lightweight, easy-to-use integration of the DeepSeek AI API for Unity projects. This package allows Unity developers to quickly implement AI-powered chat capabilities using DeepSeek's powerful language models.

Features

- Easy integration with DeepSeek API in Unity projects
- Support for both streaming and non-streaming chat completions
- Compatible with multiple DeepSeek models (DeepSeek Chat, DeepSeek Reasoner)
- Ready-to-use UI components for chat interactions
- Customizable API settings through the Unity Inspector
- Works on all platforms supported by Unity

Requirements

- Unity 2020.3 LTS or newer
- TextMeshPro package (included in newer Unity versions)
- DeepSeek API key (obtain from DeepSeek's website)

Installation

Option 1: Unity Package Manager (Git URL)

1. Open your Unity project

2. Go to Window > Package Manager
3. Click the “+” button in the top-left corner
4. Select “Add package from git URL…”
5. Enter the repository URL: `https://github.com/yagizeraslan/DeepSeek-Unity.git`
6. Click “Add”

Option 2: Manual Installation

1. Download or clone this repository
2. Copy the `DeepSeek` folder into your Unity project’s `Assets` folder

Option 3: Unity Asset Store

1. Open the Unity Asset Store in your browser or through Unity
2. Search for “DeepSeek API Integration”
3. Purchase or download the package
4. Import the package into your project

Quick Start

1. Add the `DeepSeekChat` prefab to your scene
2. Enter your DeepSeek API key in the Inspector
3. Customize the chat appearance and behavior through the Inspector
4. Press Play to start testing

Example Usage

Basic Chat Implementation

```
using UnityEngine;
using DeepSeek;

public class DeepSeekExample : MonoBehaviour
{
    [SerializeField] private string apiKey = "YOUR-API-KEY";
    private DeepSeekApi deepSeekApi;

    private void Start()
    {
        // Initialize the API
        deepSeekApi = new DeepSeekApi(apiKey);

        // Send a simple request
        SendSimpleMessage();
    }

    private async void SendSimpleMessage()
    {
        var request = new ChatCompletionRequest
        {
            Model = DeepSeekModel.DeepSeekV3.ToModelString(),
            Messages = new System.Collections.Generic.List<ChatMessage>
            {
                new ChatMessage { Role = "system", Content = "You are a helpful assistant."
                new ChatMessage { Role = "user", Content = "Hello, who are you?" }
            },
            Temperature = 0.7f,
            Stream = false
        };

        var response = await deepSeekApi.CreateChatCompletion(request);

        if (response != null && response.Choices != null && response.Choices.Count > 0)
        {
            Debug.Log("DeepSeek Response: " + response.Choices[0].Message.Content);
        }
    }
}
```

Streaming Response Example

```
private async void SendStreamingMessage()
{
    var request = new ChatCompletionRequest
    {
        Model = DeepSeekModel.DeepSeekV3.ToModelString(),
        Messages = new System.Collections.Generic.List<ChatMessage>
        {
            new ChatMessage { Role = "system", Content = "You are a helpful assistant." },
            new ChatMessage { Role = "user", Content = "Write a short story about a robot."
        },
        Temperature = 0.7f,
        Stream = true
    };

    await deepSeekApi.CreateChatCompletionStreaming(request, HandleStreamingResponse);
}

private void HandleStreamingResponse(ChatMessage partialMessage, bool isDone)
{
    // Update UI with partial response
    Debug.Log("Partial response: " + partialMessage.Content);
    if (isDone)
    {
        Debug.Log("Streaming complete!");
    }
}
```

Advanced Configuration

Available Models

The integration supports multiple DeepSeek models:

```
// Use DeepSeek Chat
var model = DeepSeekModel.DeepSeekV3;

// Use DeepSeek Reasoner
var model = DeepSeekModel.DeepSeekR1;
```

Customizing Chat UI

You can customize the appearance of the chat interface by modifying the prefab or the UI components in your scene:

1. Select the DeepSeekChat GameObject in your scene
2. Modify properties in the Inspector:
    - Chat scroll view height and width
    - Message bubble appearance
    - Input field size and position
    - Font styles and colors

Troubleshooting

Common Issues

API Key Not Working - Ensure your API key is correctly entered in the Inspector - Check DeepSeek's website to verify your API key is active

Slow Response Times - Consider using the streaming API for faster perceived response times - Check your network connection

Error Messages - "API Key is required" - Ensure you've added your API key in the Inspector - "Request failed" - Check your internet connection and API key validity

License

This project is licensed under the MIT License - see the LICENSE file for details.

Acknowledgements

- DeepSeek for their powerful AI models

Contact

- **Name**: Yağız ERASLAN
- **Email**: yagizeraslan@gmail.com
- **LinkedIn**: https://www.linkedin.com/in/yagizeraslan/

If you have any questions, suggestions, or issues, please feel free to contact me or open an issue on GitHub.