Callbacks

This project implements a modular callback system for Unity that provides easy-to-use MonoBehaviour lifecycle event hooks. It's designed to be lightweight, reusable, and easily integrated into any Unity project.

Table of Contents

Overview

The Unity Callback System provides a unified way to handle MonoBehaviour lifecycle events through UnityEvents. It offers:

- Clean separation of concerns for event handling
- Easy integration with Unity's Inspector
- Type-safe callback registration
- Minimal overhead and memory allocation

Scripts

Core

MonoBehaviourCallback.cs

The **MonoBehaviourCallback.cs** script serves as the base class for all callback components.

```
public class MonoBehaviourCallback : MonoBehaviour
{
    public CallbackEvent Callback = new();
}
```

```
[System.Serializable]
public class CallbackEvent : UnityEvent<MonoBehaviourCallback>
        { }
```

Key Features:

- Base functionality for all callback types
- Serializable UnityEvent for Inspector integration
- Type-safe callback handling

Callbacks

AwakeCallback.cs

The **AwakeCallback.cs** script handles the Awake event in the Unity lifecycle.

```
public class AwakeCallback : MonoBehaviourCallback
{
    private void Awake()
    {
        Callback?.Invoke(this);
    }
}
```

Key Features:

- Triggered once when the script instance is being loaded
- Ideal for component initialization
- Executes before Start

StartCallback.cs

The **StartCallback.cs** script manages the Start event callback.

```
public class StartCallback : MonoBehaviourCallback
{
    private void Start()
    {
        Callback?.Invoke(this);
    }
}
```

Key Features:

- Triggered once before the first frame update

- Useful for initialization that requires other components to be ready
- Executes after all Awake calls are completed

UpdateCallback.cs

The **UpdateCallback.cs** script provides frame-by-frame update callbacks.

```
public class UpdateCallback : MonoBehaviourCallback
{
    private void Update()
    {
        Callback?.Invoke(this);
    }
}
```

Key Features:

- Called every frame
- Ideal for input handling and regular updates
- Frame-rate dependent

FixedUpdateCallback.cs

The **FixedUpdateCallback.cs** script handles physics-based update callbacks.

```
public class FixedUpdateCallback : MonoBehaviourCallback
{
    private void FixedUpdate()
    {
        Callback?.Invoke(this);
    }
}
```

Key Features:

- Called at fixed time intervals
- Perfect for physics calculations
- Frame-rate independent

Getting Started

1. Add the desired callback component to your GameObject
2. In the Inspector, set up your callback events using UnityEvents
3. Create methods that accept a `MonoBehaviourCallback` parameter to handle the events

---

Usage

Basic Setup

1. Add a callback component (e.g., `UpdateCallback`) to your GameObject
2. In the Inspector, click the '+' button under the Callback event
3. Drag your target component to the event slot
4. Select your handling method from the dropdown

Example Code

```
public class ExampleHandler : MonoBehaviour
{
    public void HandleCallback(MonoBehaviourCallback callback)
    {
        // Handle the callback event
        Debug.Log($"Received callback from
         {callback.GetType().Name}");
    }
}
```

Best Practices

- Use `AwakeCallback` for component initialization
- Use `StartCallback` for inter-component setup
- Use `UpdateCallback` for frame-based logic
- Use `FixedUpdateCallback` for physics calculations

License