Unity Inspector Button

Overview

The **Unity Inspector Button Asset** is a powerful tool designed to streamline your Unity development workflow by adding custom buttons directly in the Inspector window. With this asset, you can annotate methods in your scripts with a `Button` Attribute to create easily accessible buttons in the Unity Inspector, allowing you to execute methods with a single click.

Features

- **Custom Button Attribute**: Annotate your methods with the `Button` to automatically generate buttons in the Inspector.
- **Customizable Button Labels**: Specify custom names for your buttons or use the method names by default.
- **Inspector Integration**: Buttons appear directly in the Unity Inspector for the annotated methods, making it easier to trigger functions during development.

Installation

1. **Download and Import**: Get the asset from the Unity Asset Store and import it into your Unity project.
2. **Include Namespace**: Ensure you include the SABI namespace where your attributes and editor scripts are defined.

Usage

1. **Add the Attribute to Methods**: Decorate any public or private method in your MonoBehaviour scripts with the `[Button]` attribute to create a button in the Inspector.

```
    using UnityEngine;
using SABI;

public class ExampleScript : MonoBehaviour
{
    [Button("Click Me")]
    private void MyButtonMethod()
    {
        Debug.Log("Button Clicked!");
    }
}
```

1. **Inspect Your Component**: Select the GameObject with the script attached. The method annotated with `[Button]` will appear as a clickable button in the Inspector window.

How It Works

- **Button Attribute**: The `[Button]` attribute can be applied to methods to generate buttons in the Unity Inspector.
- **Custom Inspector**: The `ButtonInspector` class extends the default Inspector and dynamically generates buttons for methods with the `[Button]` attribute. The button's label can be customized via the attribute parameter or defaults to the method name.

Customization

Custom Button Name: You can specify a custom name for the button by providing a string to the `ButtonAttribute` constructor.

```
[Button("My Custom Button")]
private void MyCustomMethod()
{
    // Method implementation
}
```

Limitations

- **Parameter Handling**: The asset does not currently handle method parameters. Methods without parameters are supported.
- **Editor Restrictions**: The custom inspector works with `Object` type and its derived classes. Ensure that the script using the attribute inherits from `MonoBehaviour` or `ScriptableObject`.

Benefits

- **Enhanced Workflow**: Quickly execute methods from the Inspector without entering play mode or writing additional editor scripts.
- **Improved Debugging**: Test and debug methods on the fly by invoking them directly from the Inspector.
- **Flexible Integration**: Easily integrate into any Unity project with minimal setup.

Getting Started

1. **Import the Asset**: After importing the asset, you will see the custom button functionality in the Inspector.

2. **Add Attributes**: Apply the `[Button]` attribute to methods in your scripts to start using the custom buttons.
3. **Use Buttons**: Select the GameObject with your script and interact with the buttons in the Inspector to execute methods.