DTT
APPS WEB GAMES

SINCE 2010

Editor Utilities

Documentation | 17-05-2022

Table of Contents

1. Get started quickly

To start using the utility tools of the asset, import the **DTT.Utils.EditorUtilities** namespace into your script file. You are now able to start using the extensive pack of tools the asset provides.

2. Introduction

The goal of the **Editor Utilities** asset is to provide users with a broad range of tools to reduce work time, optimize project code and boost workflow in the Unity Editor. It contains one namespace containing a multitude of editor-only scripts.

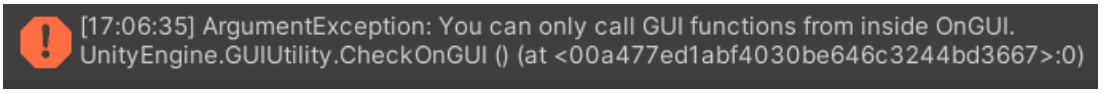
3. Set-Up

Once you have imported the **DTT.Utils.EditorUtilities** into your script file, you can start working with the multitude of classes the asset provides.

4. API

4.1 Caching GUI Content and GUI Styles

An issue you have most likely already encountered when creating your own GUIStyle and GUIContent fields is that they can't be created outside of an OnGUI method.



This makes editor scripting unnecessarily a lot more difficult. The asset provides two classes that focus on fixing this issue.

```
#if UNITY_EDITOR

using System;
using DTT.Utils.EditorUtilities;
using UnityEditor;
using UnityEngine;

public class CustomWindow : EditorWindow
{
    /// <summary>
    /// Derives from the GUIStyleCache to delay creation of styles
    /// to the first time they are used.
    /// </summary>
    private class Styles : GUIStyleCache
    {
        /// <summary>
        /// The style for header labels.
        /// </summary>
        public GUIStyle Header => base[nameof(Header)];

        /// <summary>
        /// Initializes the cache with a style for header labels.
        /// </summary>
        public Styles()
        {
            // The first time the style is used, its instance will be created.
            Add(nameof(Header), () => new GUIStyle(EditorStyles.label)
            {
                alignment = TextAnchor.UpperRight,
                fontSize = 15
            });
        }
    }

    /// <summary>
    /// The styles instance which holds the custom styles.
    /// </summary>
    private readonly Styles _styles = new Styles();

    /// <summary>
    /// Draws a label using the custom style.
    /// </summary>
    private void OnGUI() => GUILayout.Label("Title", _styles.Header);
}

#endif
```

4.2 Interacting with the Asset Database

```
private void OnEnable()
{
    // Load a component attached to a prefab with precise error messaging when failing.
    string prefabPath = "Assets/DTT/GameObject.prefab";
    TestScript script =  AssetDatabaseUtility.GetComponentInPrefab<TestScript>(prefabPath);

    // Load a scriptable object but create it if it could not be found.
    string objectPath = "Assets/DTT/ScriptableObject.asset";
    TestSO scriptableObject = AssetDatabaseUtility.GetOrCreateScriptableObjectAsset<TestSO>(objectPath);

    // Load all TestSO scriptable objects in the project.
    TestSO[] scriptableObjects = AssetDatabaseUtility.LoadAssets<TestSO>();
}
```

4.3 Creating custom popup windows

```
private void OnEnable()
{
    // Create a builder to show a contextual menu on the screen.
    ContextDropdownBuilder builder = new ContextDropdownBuilder();

    // Use "StartIndex" to created nested items.
    builder.StartIndent("Create");
    builder.AddItem("GameObject");
    builder.AddItem("Scriptable Object");
    builder.EndIndent();

    // Execute a custom method when your item has been clicked.
    builder.AddItem("Cool Behaviour", DoSomethingCool);

    // Show your popup when ready.
    builder.GetResult().Show();
}
```

4.4 Remaining API

The above covers the Editor Utilities API you are most likely to use. This, however, is not an exhaustive list and you can find further functionality by exploring the namespace. If you have any questions regarding one of the undocumented methods or classes, feel free to contact us!

5. Known Limitations

This asset currently does not have any known limitations.

6. Support and feedback

If you have any questions regarding the use of this asset, we are happy to help you out. Always feel free to contact us at:

unity-support@d-tt.nl

(We typically respond within 1-2 business days)

We are actively developing this asset, with many future updates and extensions already planned. We are eager to include feedback from our users in future updates, be they 'quality of life' improvements, new features, bug fixes or anything else that can help you improve your experience with this asset. You can reach us at the email above.

Reviews and ratings are very much appreciated as they help us raise awareness and to improve our assets.

DTT stands for Doing Things Together

DTT is an app, web and game development agency based in the centre of Amsterdam. Established in 2010, DTT has over a decade of experience in mobile, game, and web based technology.

Our game department primarily works in Unity where we put significant emphasis on the development of internal packages, allowing us to efficiently reuse code between projects. To support the Unity community, we are publishing a selection of our internal packages on the Asset Store, including this one.

More information about DTT (including our clients, projects and vacancies) can be found here:

https://www.d-tt.nl/en/