Table of Contents

Quick Start ---------- 2

Quick Start

This plugin provides two classes, RALogHandler and TaggedRALogger.

RALogHandler

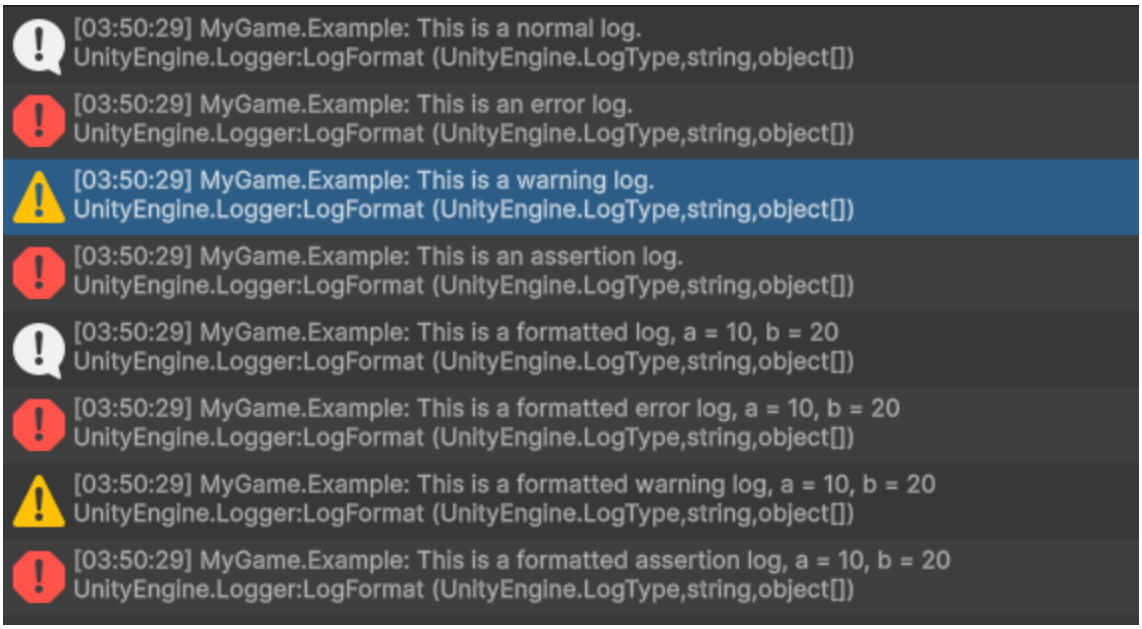
`RALogHandler`, or "Release-Aware" Log Handler, is an implementation of Unity's `ILogHandler` which only logs in the Editor, development builds, and builds with the custom scripting symbol `ULOGGERS_ALWAYS_LOG` defined. You can use this just like any other implementation of `ILogHandler`.

TaggedRALogger

`TaggedRALogger` is a release-aware, tagged alternative to `Debug.Log` and its variants. By default, it uses Unity's log handler (`Debug.unityLogger.logHandler`), but that can be overriden in its constructor.

```
using UnityEngine;
using Uralstech.Utils.Loggers;

namespace MyCompany.MyGame
{
    public class Example : MonoBehaviour
    {
        private static TaggedRALogger s_logger = new($"{nameof(MyGame)}.{nameof(Example)}");

        private void Start()
        {
            s_logger.Log("This is a normal log.");
            s_logger.LogError("This is an error log.");
            s_logger.LogWarning("This is a warning log.");

            s_logger.Log(LogType.Assert, "This is an assertion log.");

            int a = 10, b = 20;
            s_logger.Log("This is a formatted log, a = {0}, b = {1}", a, b);
            s_logger.LogError("This is a formatted error log, a = {0}, b = {1}", a, b);
            s_logger.LogWarning("This is a formatted warning log, a = {0}, b = {1}", a, b);

            s_logger.Log(LogType.Assert, "This is a formatted assertion log, a = {0}, b =
{1}", a, b);
        }
    }
}
```

The output of the above script looks like this:



Breaking Changes Notice

If you've just updated the package, it is recommended to check the *changelogs* for information on breaking changes.