

Static State Auditor

©Copyright Digital Software/Johan 'Zarkow' Munkestam

Version: 1.0.0 (Unity 6.3 LTS, 6000.3 and newer; editor-only, any render pipeline)

Introduction

When "Enter Play Mode" runs without a domain reload, play mode starts in well under a second. It also stops resetting your static fields: singletons keep pointing at objects from the previous run, static events keep their subscribers, static counters continue where they stopped. Unity's guidance is to reset every static yourself in a method marked `[RuntimeInitializeOnLoadMethod(RuntimeInitializeLoadType.SubsystemRegistration)]`. Static State Auditor lists every static in your own assemblies so you know what to reset.

Getting started

Open Window > Digital Software > Static State Auditor and press Scan. The list shows one row per static field, static auto-property and static event found in the assemblies compiled from your Assets folder.

Demo: open the scene under Assets\DigitalSoftware\StaticStateAuditor\DemoScene, enable Edit > Project Settings > Editor > Enter Play Mode Settings with "Reload Domain" unticked, and press Play twice. The run counter on screen climbs and the console warns about a stale singleton; the AlreadyReset component behaves the same on every run because it resets its statics.

The window

Scan: reflects over your assemblies again. Press it after every compile.

Include packages: also lists assemblies under Packages and the package cache. Off by default.

Editor code: shows statics declared in editor-only assemblies and in editor namespaces inside runtime assemblies (packages often wrap editor types in defines). They are editor state, not play-mode state, so they are hidden by default.

Filter: matches type, member or assembly names.

Classification toggles: mutable, event, reference, constant. Constants are hidden by default.

Status line: reads your Enter Play Mode settings and says what they mean for the rows below; the button opens the Editor settings page.

Double-click a row to open the script at the declaration. Click a column header to sort.

Export CSV: writes the current view, with a header of # lines that records the project, Unity version, scan time, settings and totals.

Classifications

Mutable: a static field, or a static auto-property, that is not readonly. It keeps its value between play sessions. Needs a reset.

Event: a static event. Subscribers added in one play session are still subscribed in the next, so handlers run twice, three times, and so on. Needs a reset (set the event to null).

Reference: a readonly field of a reference type, for example a static readonly List. The reference cannot change but the contents can. Review it: clear it in a reset method if it is filled during play.

Constant: a readonly value type, string or delegate instance. Cannot change after type initialisation. Safe. Const fields are never listed because they are compile-time values.

Reset column: "yes" when the declaring class, or a class that encloses it, has a static method with `RuntimeInitializeOnLoadMethod(SubsystemRegistration)`. "other load type" when the attribute uses a different load type, which runs later than `SubsystemRegistration` and may be too late for code that touches the static during scene load.

Fixing a static

Add a static method to the class and reset every mutable member in it:

```
[RuntimeInitializeOnLoadMethod(RuntimeInitializeLoadType.SubsystemRegistration)]
```

```
private static void ResetStatics()
```

```
{
```

```
    _instance = null;
```

```
    SpawnCount = 0;
```

```
OnMessage = null;  
}
```

Unity calls it before the first scene loads, in every play session, with or without domain reload. Rescan and the row shows "yes" in the Reset column. Where a static only ever holds a constant, mark it readonly instead and it drops out of the mutable list.

Pro

Static State Auditor Pro adds a generated reset file with one SubsystemRegistration method per class, a play-mode diff that lists which statics changed and which survived exit, an ignore list with reasons, and stub insertion into your own classes. Upgrade pricing from this free package is set in the store.

Further support

Head over to <https://community.digitalsoftware.se/> or the Discord at <https://discord.gg/fyqtxZt>