Property Inspector

v: 1.0.0.1

by: Matheus de Almeida - temdisponivel@gmail.com |
github.com/temdisponivel

Property Inspector is a tool that enhance your productivity by allowing you to search and edit any property inside a object - or multiple objects (with multi-editing).

Search:

Once you've selected what objects you want to edit, be from hierarchy, project or both, you can search a property by its name using several methods: starts with - where properties whose names starts with the search will be shown; ends with - where properties whose names ends with the search will be shown; and match - where properties whose name match exactly the search will be shown. If you type multiple words separated by a space, only properties whose names contains all those words will be shown. All these methods are NOT case sensitive.

When you type a search, all properties whose name contains the query will be shown. Even deeply nested properties.

You can also search a property by its type, allowing you to show only properties of type "Player", for example.

Another option is the search by path, where you type the path of the property you want to see. For example: Player.HealthHandler.Life would only show the property Life of the property HealthHandler of the property Player. This is specially useful when you want to edit lists, where you can search for a specific index, using <ListName>.Array.data[<index>].

Multi-edit:

You can multi-edit objects using the option "Multi-edit". When multi-editing, all objects - and components - with the same type will be grouped together allowing you to edit them as if they were one. If you change any property using multi-editing, all objects of that group will be affect.

Inspector mode:

The inspector mode is used when there's no search query. If it's checked, all properties from all selected objects will be shown, much like Unity's Inspector, hence "Inspector mode".

Selection history:

The selection history enables you to navigate to previous or next selections. It saves all selections made when the Property Inspector was seen by you.

Custom Inspectors:

The custom inspector mode allows you to use yours custom inspectors to draw objects/components. In this mode, you will see basically the same thing you would see inside Unity's Inspector. The "Inspector mode" should be on in order for this to work, since if it's not on, there will be no data to display.

Apply/revert:

In order to streamline your workflow, there's also an apply and revert button in all objects/components that are instances of prefabs. These buttons will apply or revert the changes you've made to those objects/components. When multi-editing, the changes will be will affect all prefabs linked to those instances.

Edit script:

If you've any asset defined by a script - scriptable object instances, game objects with components, etc - the "Edit script" button will be enable so that you can quickly open the script that defines this asset.

Highlight:

There's also a highlight button when in single editing (not in multi-editing). This button will highlight the object being edited in the hierarchy/project.

When you double click this button, it will select the object in your hierarchy or scene (or potentially both in the case of multi-edit).

Lock:

There's a lock option that allows you to lock the current selected objects so that you can continue to work on other things while keeping the selection for when you come back later. It works exactly the same as the lock option in Unity's Inspector.

Window types:

You can use two types of windows, a dockable one that you can drag and dock and a popup one. The later is useful for quick edits since it can be open using ALT + F shortcut and closed pressing ESCAPE.

Undo:

All changes (except apply and revert) made inside Property Inspector can be undone using CTRL + Z (CMD + Z).

Shortcuts:

ALT + T open Property Inspector in popup mode.

Notes:

It's worth noting that this tool is not suppose to replace Unity's inspector, but rather to extend it and work aside it to enhance and streamline your workflow.

If you have any question, ran into bug or problem or have a suggestion please don't hesitate in contacting me at: temdisponivel@gmail.com.

Credits:

Developed by Matheus de Almeida.

Huge thanks to Allan Smith and Fire Horse Studios.