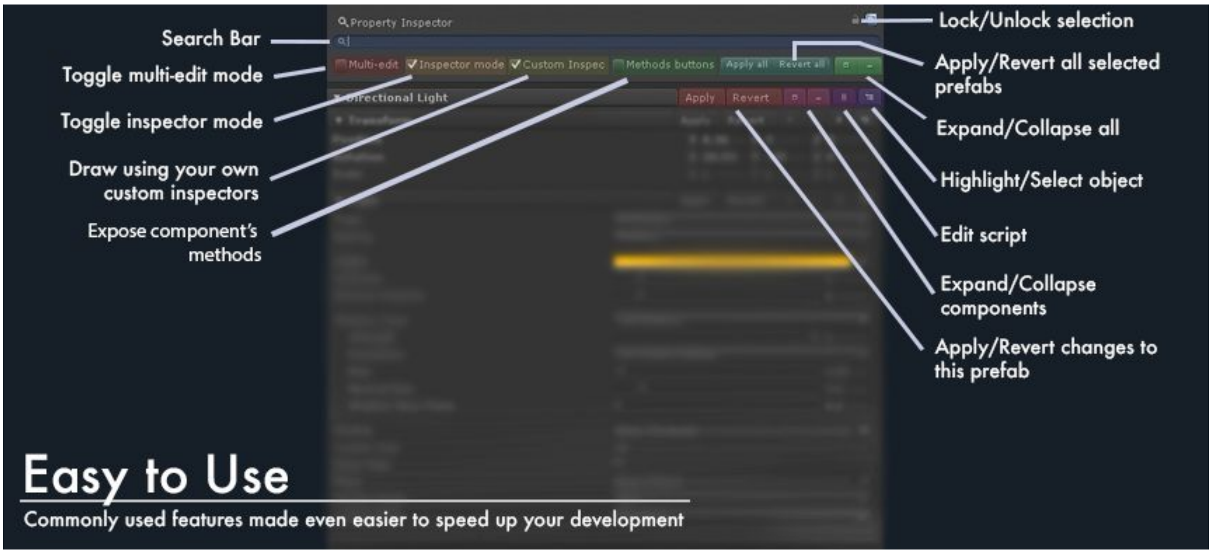
Property Inspector

v: 1.0.0.4

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.



Open:

You can open Property Inspector through Window -> Property Inspector Popup or Window -> Property Inspector Window.

You can also use the shortcut ALT + F to open as a popup.

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 patterns:

- Plain search (without prefixes) will show properties whose names contains any of the parts of the typed text (parts being separated by spaces);
- Starts with (prefix: s:) will show properties whose names starts with the typed text;
- Ends with (prefix: e:) will show properties whose names ends with the typed text;
- Type (prefix: t:) will shows properties whose type name match the typed text. For example: "t:string" will show all strings; "t:Player" will show all players.
- Value (prefix: v:) will show properties whose value match the typed text. Valid for strings, integers, floats and booleans. Some of Unity classes, such as Vectors and Color, are supported as well, but you'll have to type the same value as would be shown if ToString() was called on the property.
- Match (prefix: m:) will show all properties whose name exactly match the typed text.
- Path (no prefix needed): If you type a path (using dots) like Player.Health, the plugin will search for a property with that path. So, if you have a property of type player

called MyPlayer, that has a property of type float called Health, you can type “MyPlayer.Health” and the plugin will show you only the health property of the MyPlayer property.

Tips:

All these search will iterate through properties and their children, so even deeply nested properties will be found.

You can filter list (or array) elements using: “ListName.Array.data[DesiredIndex]”. Where DisiredIndex is the index of the list you want to see. You can also see the list size with “ListName.Array.size”.

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.

Method buttons:

When you have the “Method buttons” options checked the Property Inspector will create a section by the end of the component’s inspector with a bunch of buttons to allow you to call method without the need to create custom inspectors. All parameter-less methods will produce a button. When you call a method of a group, the method will be called on all objects from that group.

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 that will highlight the clicked object on the hierarchy/project window. When you click the highlight button of a group, the selection will be filtered to only that specific group.

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.