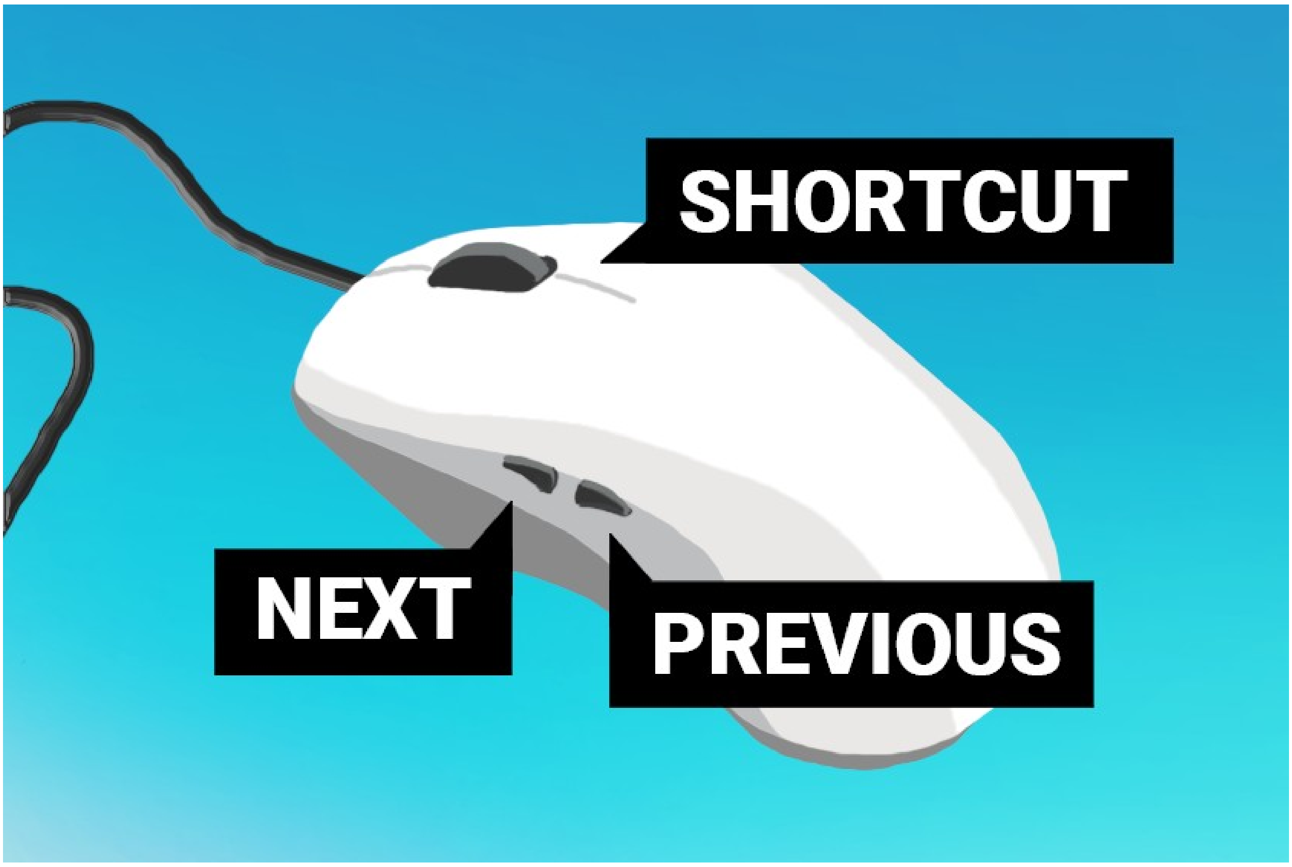
# Mouse Shortcuts - Manual



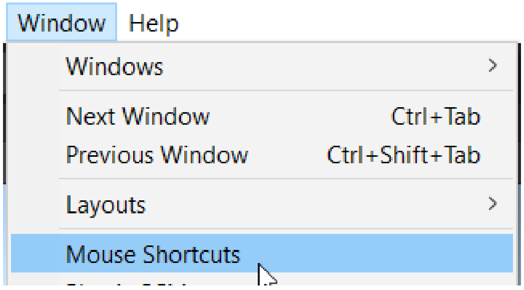
A) It allows you to define shortcuts for the 4th and 5th mouse button.

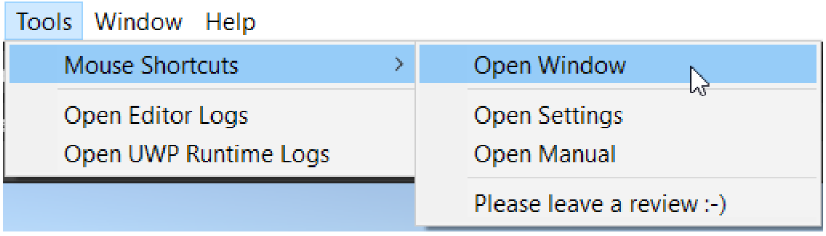
B) Adds some "selection history" commands which allow you to use the mouse buttons to go back and forth in Unity, just like in your web browser.

## Table of contents

# 1. Usage

After the installatin a new menu will be available under **Tools > Mouse Shortcuts.** From there you can open the shortcuts window, the settings and the manual.

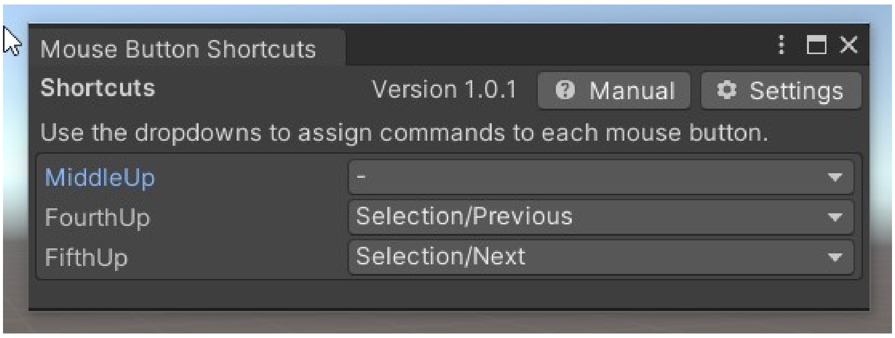




The shortcuts window allows you assign commands to your mouse buttons. The list is populated from all the methods which have a [ShortcutAttribute], a [MenuItem] attribute or a [MouseCommand] attribute.

Choose „-“ from the dropdown to have no command executed.

By default the 4th and 5th mouse buttons are configured to execute the custom selection history commands.

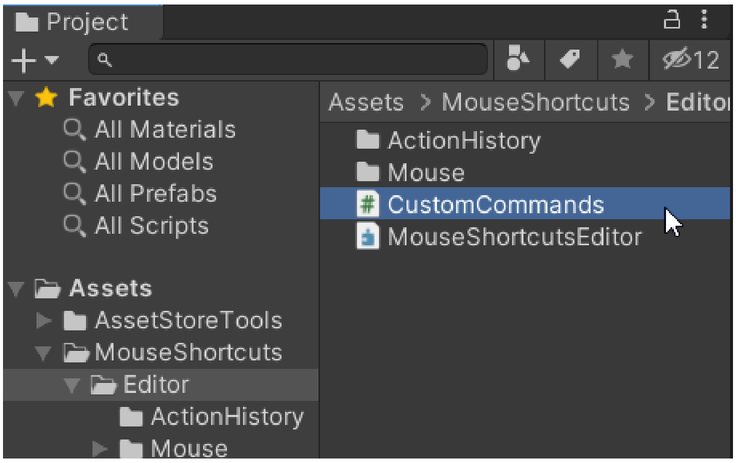


Use the dropdowns to assign commands to each mouse button.

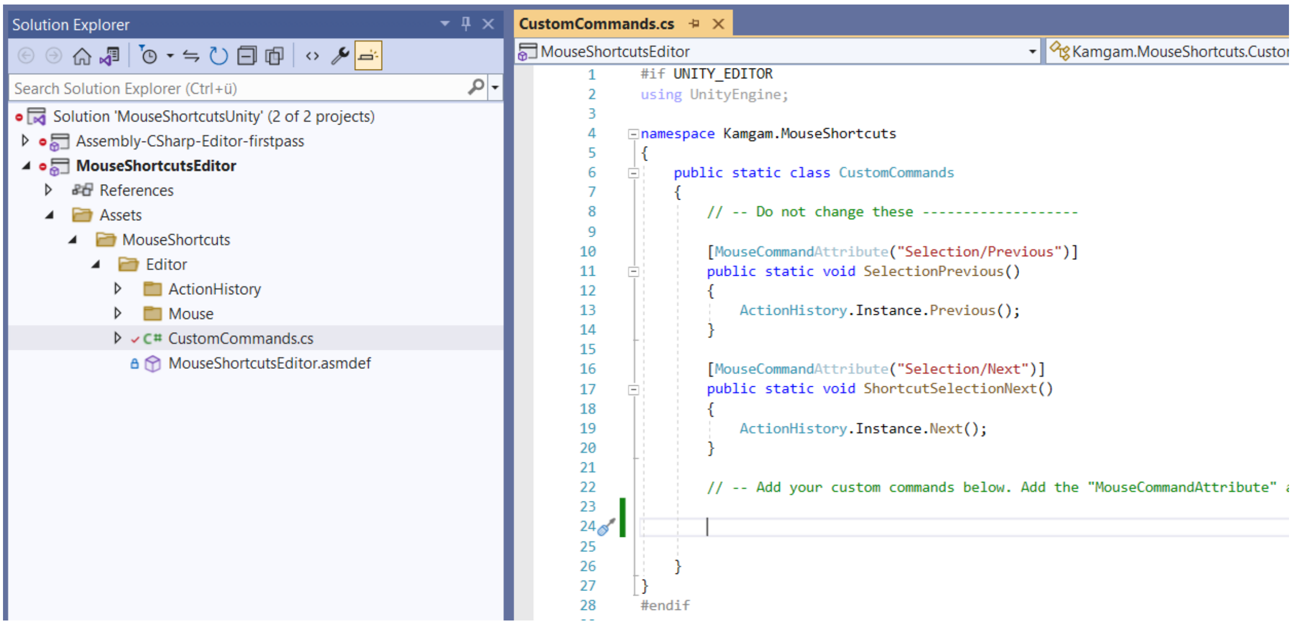
MiddleUp: -
FourthUp: Selection/Previous
FifthUp: Selection/Next

# 2. Adding custom commands

Go find the CustomCommands.cs file in your project. It’s usually located under **Assets/MouseShorcuts/Editor/CustomCommands.cs**.

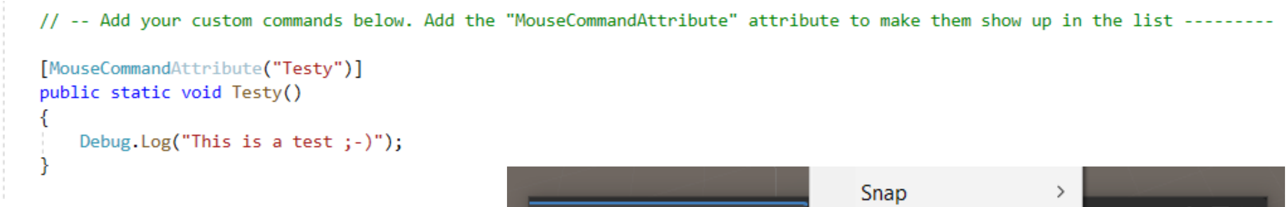


As you can see it already contains some custom methods.


Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution 'MouseShortcutsUnity' (2 of 2 projects)
Assembly-CSharp-Editor-firstpass
MouseShortcutsEditor
References
Assets
MouseShortcuts
Editor
ActionHistory
Mouse
CustomCommands.cs
MouseShortcutsEditor.asmdef
CustomCommands.cs
MouseShortcutsEditor
Kamgam.MouseShortcuts.Custor

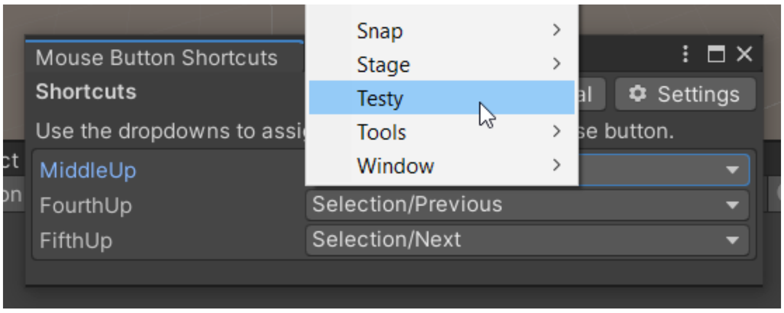
```
#if UNITY_EDITOR
using UnityEngine;

namespace Kamgam.MouseShortcuts
{
    public static class CustomCommands
    {
        // -- Do not change these ------------------

        [MouseCommandAttribute("Selection/Previous")]
        public static void SelectionPrevious()
        {
            ActionHistory.Instance.Previous();
        }

        [MouseCommandAttribute("Selection/Next")]
        public static void ShortcutSelectionNext()
        {
            ActionHistory.Instance.Next();
        }

        // -- Add your custom commands below. Add the "MouseCommandAttribute" a

        

    }
}
#endif
```

To add one on your own simply copy one and rename it.



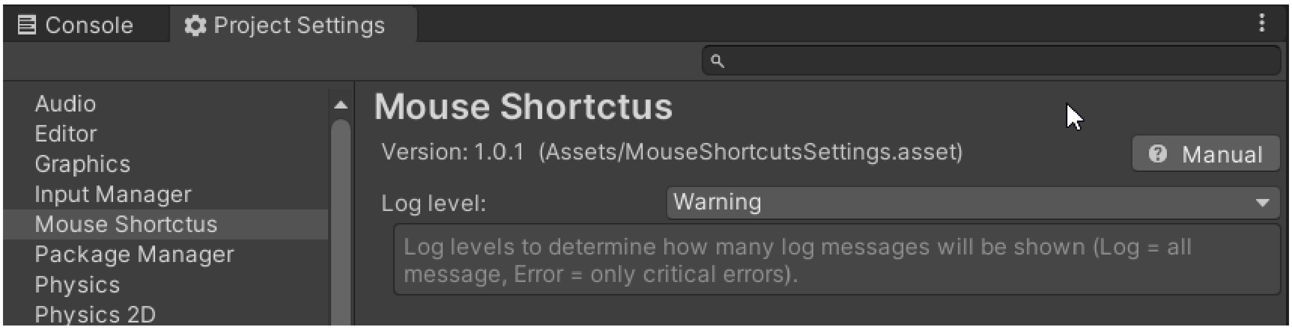
```
// -- Add your custom commands below. Add the "MouseCommandAttribute" attribute to make them show up in the list ---------

[MouseCommandAttribute("Testy")]
public static void Testy()
{
    Debug.Log("This is a test ;-)");
}
```

You can then pick it from the dropdown in the shortcuts window.



# 3. Settings

The settings are stored in a Scriptable Object located under Assets/. You can access them via the „Project Settings“ menu or via **Tools > MouseShortcuts > Settings**.



Mouse Shortctus

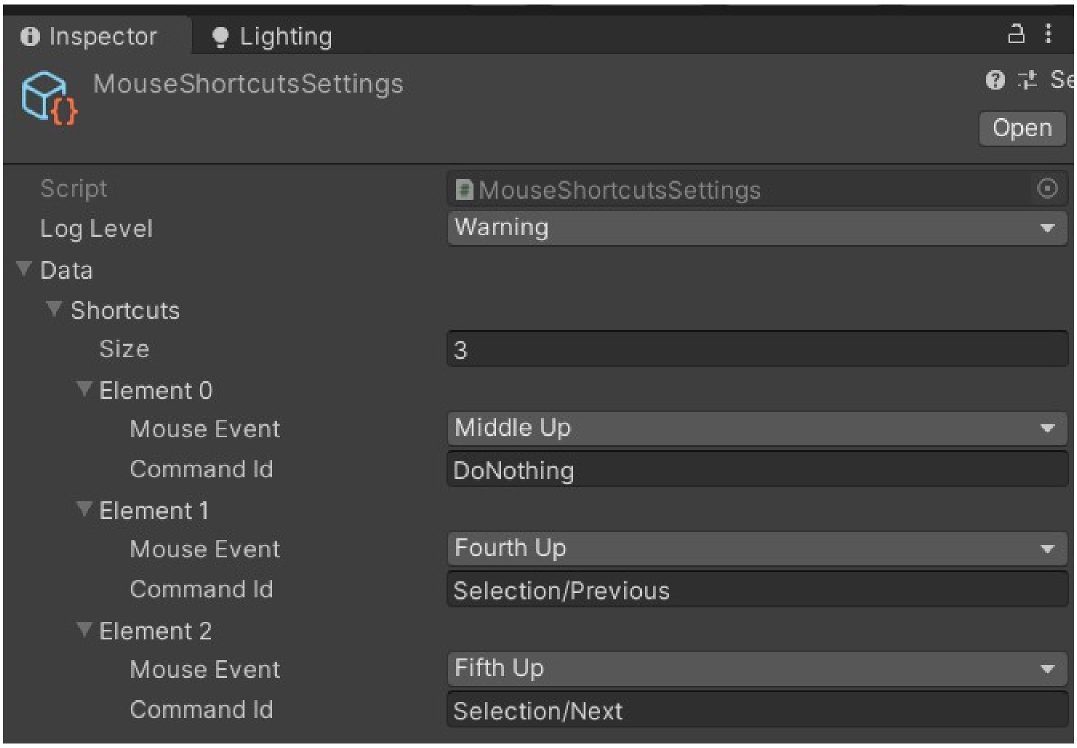
Version: 1.0.1  (Assets/MouseShortcutsSettings.asset)

Log level: Warning

Log levels to determine how many log messages will be shown (Log = all message, Error = only critical errors).

The inspector of the settings shows some more data. This is where the button-to-command assignments are stored. You can edit them manually, though it is not recommended.

Be aware that those CommandId strings have to match exactly.



Script: MouseShortcutsSettings

Log Level: Warning

Data

Shortcuts

Size: 3

Element 0

Mouse Event: Middle Up

Command Id: DoNothing

Element 1

Mouse Event: Fourth Up

Command Id: Selection/Previous

Element 2

Mouse Event: Fifth Up

Command Id: Selection/Next