W_MOVEMENU

Wouter Gilsing

USER GUIDE

version 1.0

Introduction

Hi there, thanks for trying out W_moveMenu!

W_moveMenu is a little tool to reorganize your toolbar by grouping items together and by moving items from one menu into another. The items will be removed from their original location and reappear in a location defined by the user.

Having several plugins installed, or working for a studio with a bunch of custom tools, you might you might find yourself using a Nuke with an interface packed with extra icons and menus. Sometimes those extra menus might take up quite a lot of space while not being used particularly often. There might be no room left for a user's personal tools, or the toolbar might simply appear kinda messy and distracting.

Deleting is not a good idea however, so with W_moveMenu you now have the option to simply move them instead.

After being moved, all items will preserve their shortcuts and icons, and menus will preserve all of their contents and submenus. Item will remain accessible through the tab menu, so everything will continue to work as expected.

If you encounter any bugs, you can report them by sending an email with a description of your problem to woutergilsing@hotmail.com.

Installation

1 Copy ‘W_moveMenu.py' to a folder that's part of the nuke plugin path (for example ~/.nuke).

2 Append menu.py with the following code:

```
import W_moveMenu
```

You then simply call the moveMenus method and tell it which menu items are to be moved and whereto. Add that line to you menu.py as well, as you most likely want them to happen during start up.

Arguments

Calling the moveMenu method works as follows:

```
W_moveMenu.moveMenus(
['list','of','menu items','to', 'move'], 'destination menu', 'icon',
'sourceRoot', 'destinationRoot', remove, contentsOnly
)
```

The function expects several arguments:

- The list of of the menus and/or menu items that are to be moved.
- A destination menu. (optional)
- An icon, in case the destination menu does not exist yet. (optional)
- The root menu or toolbar the items are currently stored in. By default set to ‘Nodes’. (optional)
- The root menu or toolbar the destination menu is or will be located in. By default set to the same as the sourceRoot argument (optional)
- Remove. Whether or not the items will be removed from their original location after being moved. By default set to True. (optional)
- ContentsOnly. If the list of items contains any menus, setting this value to True will make sure to ignore those menus and to skip directly to their contents instead. By default set to False. (optional)

Optional arguments do not have to be specified in case the default value needs to be used.

The following pages will feature examples of how the tool can be used, and how those arguments will affect the way the items are moved.

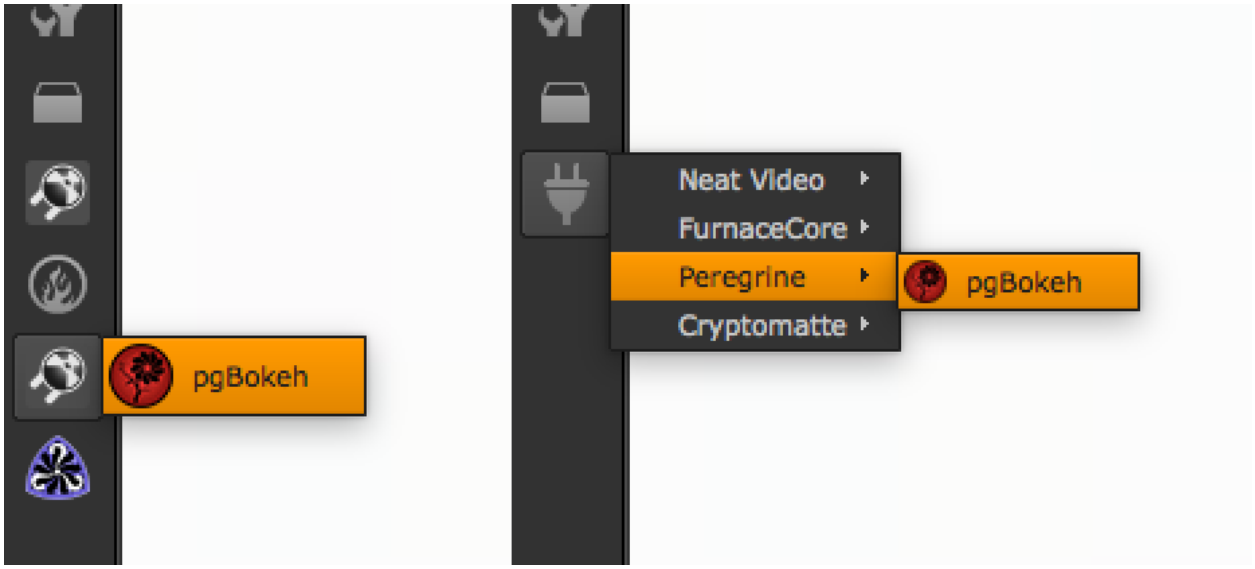
Example scenarios

Grouping menus together into a new menu

In the following example the function is used to group several menus containing plugins together into a new menu called 'Plugins'.

```
W_moveMenu.moveMenus(['Neat Video','FurnaceCore','Peregrine','Cryptomatte'],'Plugins','plugins.png')
```

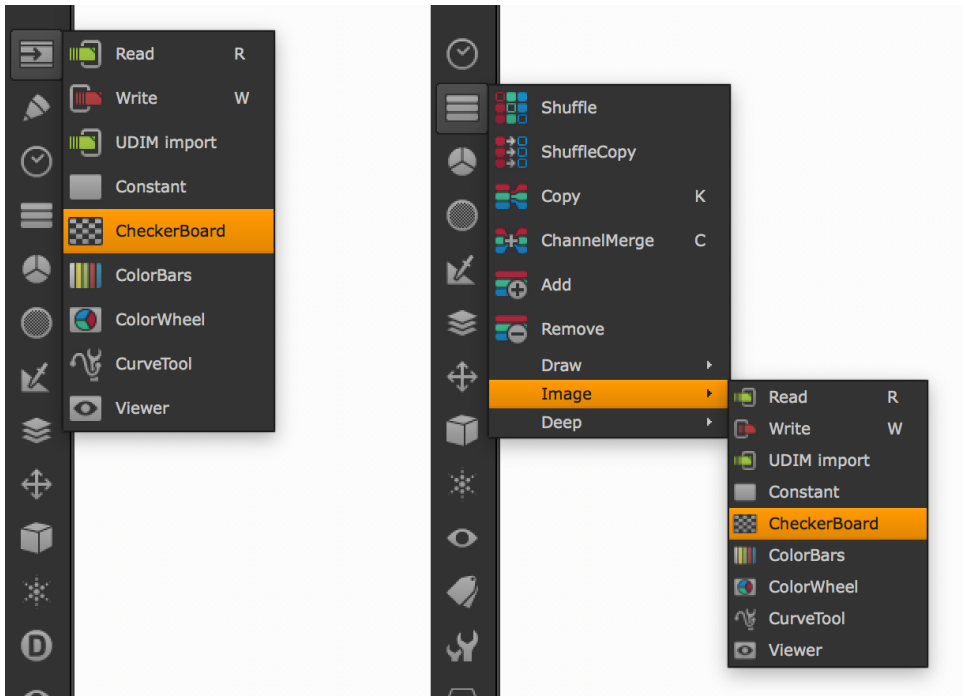
- The first argument needs to be the list of the names of menus that the user wants to move. In this case '**Neat Video**', '**FurnaceCore**', '**Peregrine**' and '**Cryptomatte**'. In case multiple items are moved, they need to be separated by comma's and surrounded by square brackets (Pythons 'list' syntax).
- The second argument is the destination menu. In this case we want to move the menus into a menu called '**Plugins**'. Since there is no menu called '**Plugins**' yet, a new menu will be created.
- Since the destination menu will be freshly created, we can define an icon to assign to it. Any icon placed in a folder being part of Nuke's plugin path will work. (So that includes the icons stored in Nuke's installation folders, like Transform.png, Grade.png, Invert.png, CornerPin.png etc.). The 'plugins.png' was a custom icon. If you want to source an image from an other location, you'll need to supply its full path.



Appending existing menus

Rather than creating a new menu, it's also possible to append any existing menus. This example will move Nuke's '**Draw**', '**Image**' and '**Deep**' menus into the '**Channel**' menu:

```
W_moveMenu.moveMenus(['Draw','Image','Deep'],'Channel')
```

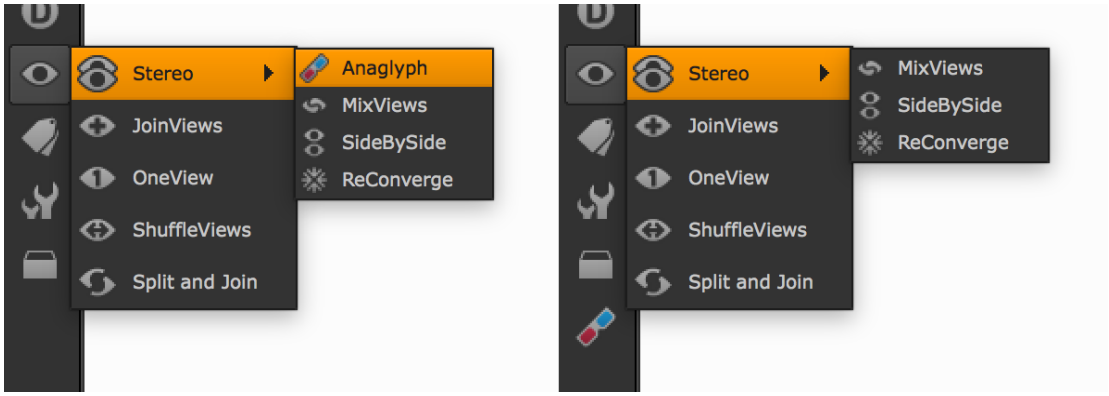


Moving items directly into the root menu

The destination argument is optional. Whenever not defined, the specified menus and menu items will be placed directly into the 'destination root' (which is by default set to mimic the 'source root', which is by default set to 'Nodes').

Note that since only one item is being moved in this example, it is not necessary to still define it as a list; a single string without the square brackets will suffice too.

```
W_moveMenu.moveMenus('Views/Stereo/Anaglyph')
```

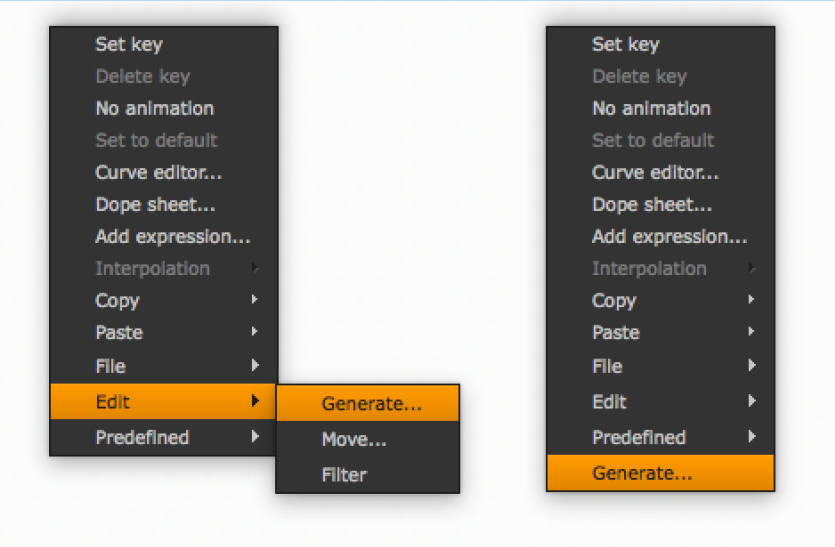


Other toolbars and menus

By default the ‘sourceRoot’ argument is set to ‘Nodes’, representing the Nodes toolbar. However, other toolbars and menus such as ‘**Nuke**’ (menu bar at the top) and context menus such as ’**Viewer**’ and ‘**Properties**’ will also work.

The following example will move the ‘Generate…’ item of the ‘**Animation’** menu. In this case we want the item to move from its submenu directly into the root menu, so there is no need to specify a destination menu.

```
W_moveMenu.moveMenus('Edit/Generate...', sourceRoot = 'Animation')
```



Cross toolbars

It is also possible to move menu items between different toolbars. In that case the user should define the ‘destinationRoot’ argument. The following example moves the ‘Keyer’ menu from the ‘Nodes’ toolbar to Nuke’s application menubar.

Setting the ‘remove’ argument to False will ensure that the menu item will remain accessible from their original location as well.

```
W_moveMenu.moveMenus(‘Keyer', destinationRoot = 'Nuke', remove = False)
```