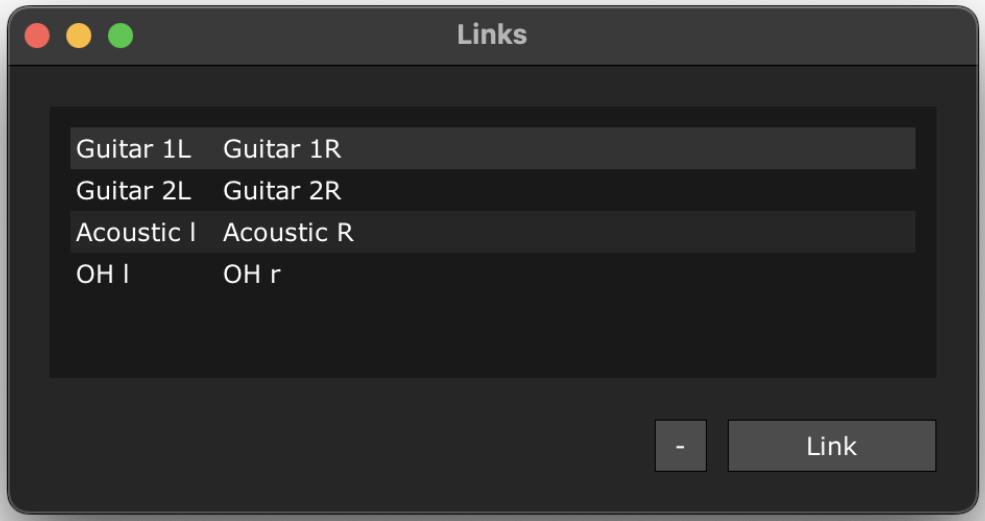
ReaLink

Marini Mattia

2023/2024



A GUI based lua script to link FX parameters across different tracks

---

This plugin runs a backgound task that links the selected tracks, ensuring that the plugin configurations of the 2 FX chains match

1 Introduction

Features

- Linking every parameters of every plugin across different tracks
- Linking state is saved on project basis
- GUI based link managment
- Multiple tabs support and hot project reloading
- Flexible and roboust linking
- Zero dependency script

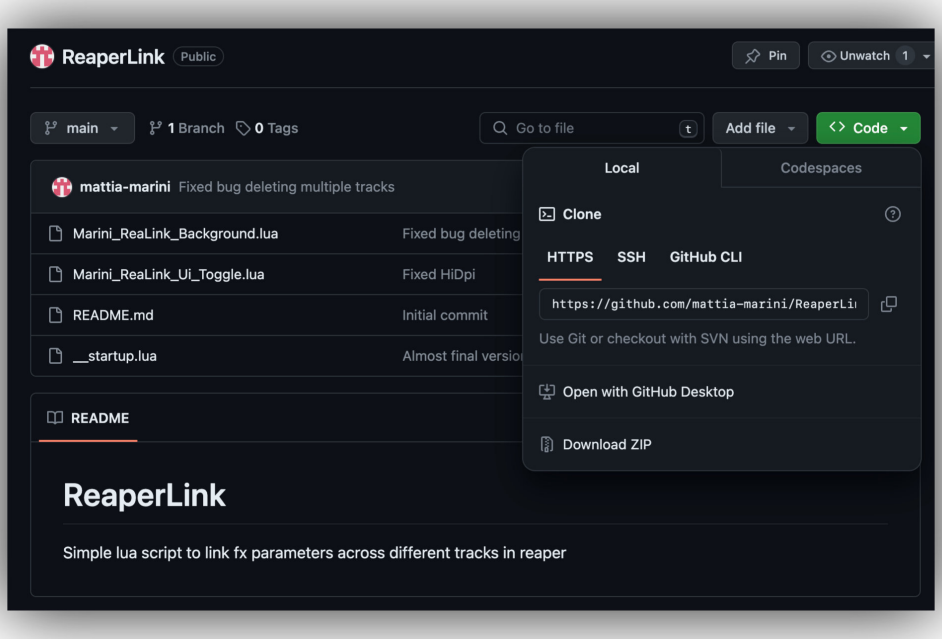
Non-features

- Single parameter or single plugin link
- Does not support undo tree (cmd/z)
- Gui does not support docked state yet
- Parameters linking on more than 2 tracks (have 1 track control 2 or more other)
- Windows HiDpi support(not testet, and highly doubt the gui would scale correctly)

2 Installation

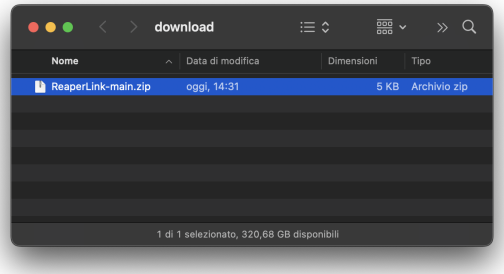
The installation works as any other reaper script. The only note is that a __startup.lua script should be added to automatically run the background synching task as soon as reaper is launched

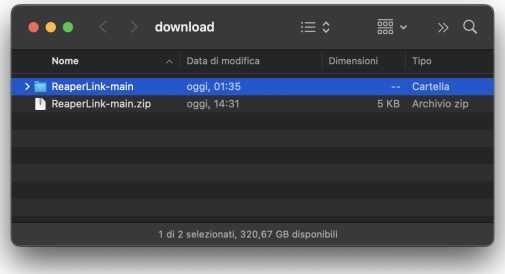
- First of all, download the script from the github repo: *https://github.com/mattia-marini/ReaperLink*.



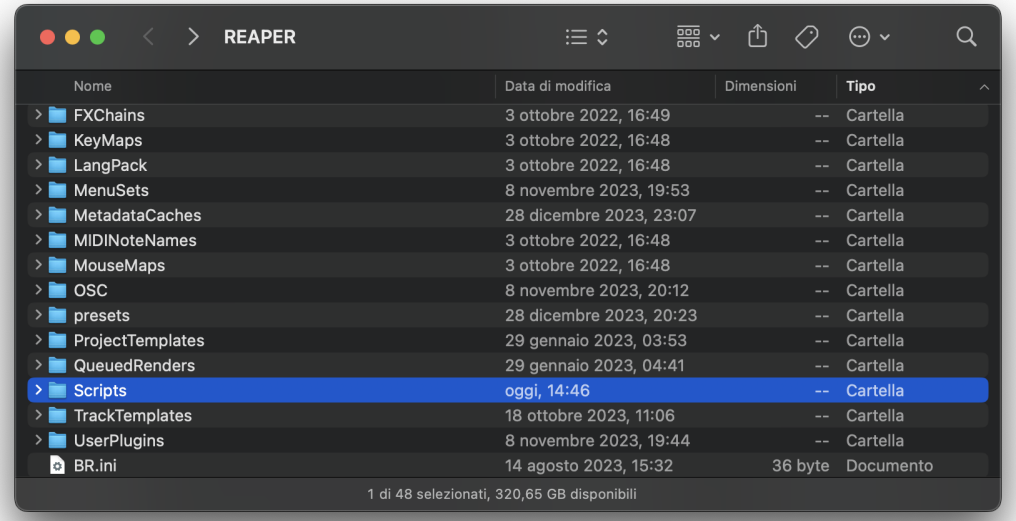
Click on *Code* and then *Download ZIP*

- It will download a zip file. Unzip it



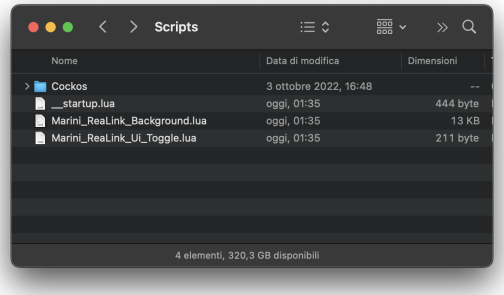


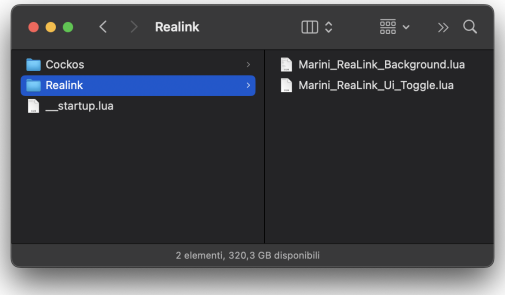
- Now we need to put the script in the correct position, so that it will be backed up with the rest of the reaper config. Open up reaper, on the menu bar go to *Options* → *Show REAPER resource path in explorer/finder*. That should open the folder containing all the reaper data



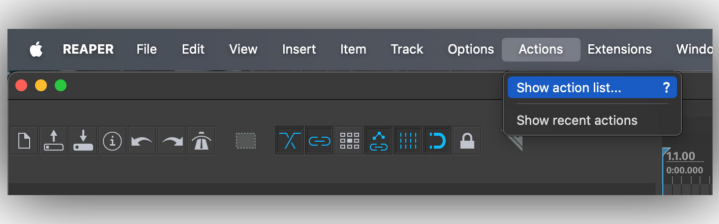
- Now you have to copy the files you just downloaded into this folder. Note that you can organize them however you want (putting them in subfolders), but the *__startup.lua* file should alway be inside the script folder[1] (you can't put it in subdirectories). The 2 more logical configurations are the following:

[1] __startup.lua is a special script that runs on the startup. We use this to automatically start the linking tast at whenever we open a new project. It has to be inside the script folder because thats the way reaper recognize it

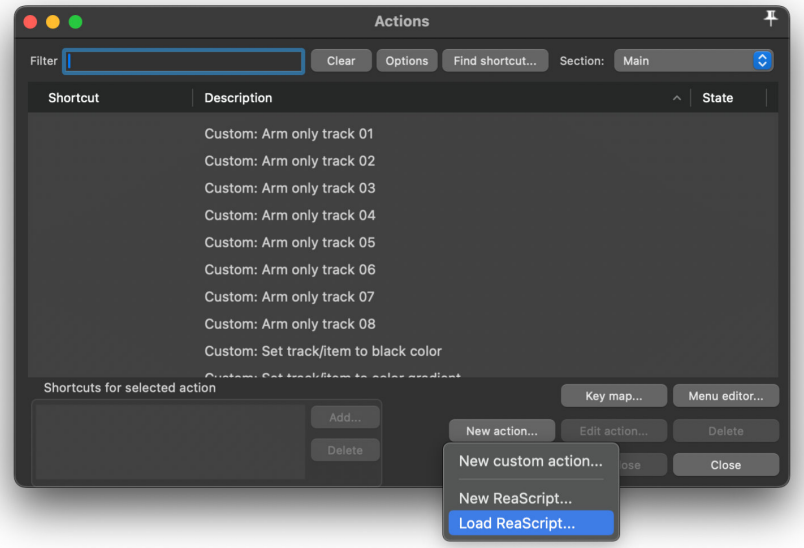




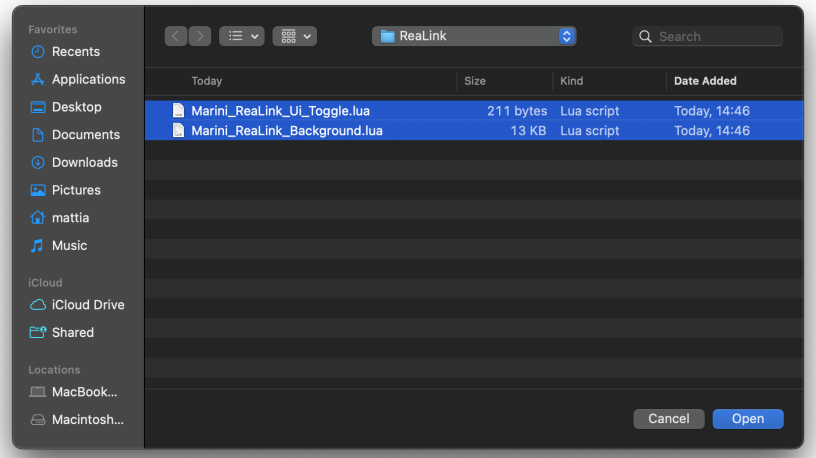
- To add the script in reaper go to *menu bar* → *Actions* → *Show action list...*



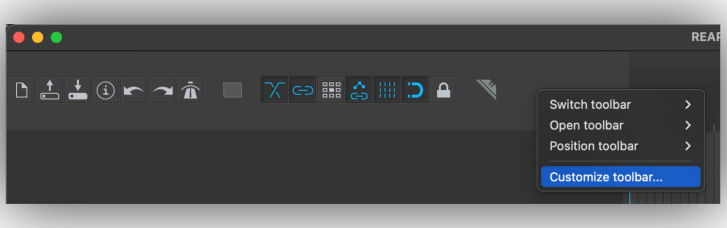
- On the window that pops up, click *New action* → *Load ReaScript...*



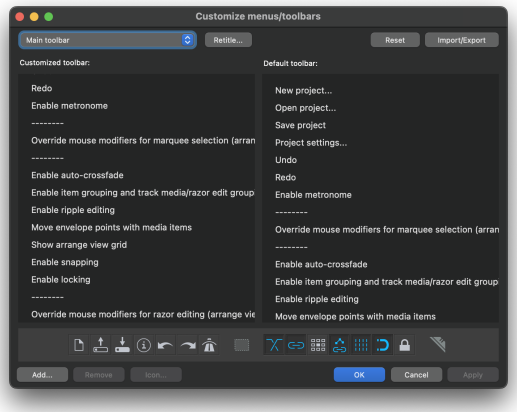
- Now navigate to the script folder, and select the 2 script files and hit *open* (you don't need to add the __startup.lua too)

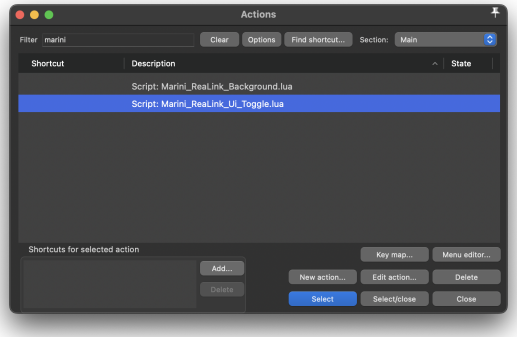


- Now its time to add a toolbar entry to toggle the UI. Go to the *main toolbar* → *right-click it* → *Customize toolbar...*:



- On the bottom left of the window that pops up click *add* and search for the action "*Marini_ReaLink_Ui_Toggle.lua*". Double click on it and apply the changes.





You should now see a new toolbar item.

- Restart reaper and you should be ready to go!

3 How to use

Linking

- Open the UI with the toolbarButton that you just set up (or by running the *Marini_ReaLink_Ui_Toggle.lua*)
- Select the 2 tracks that you want to link
- Hit *"link"* on the script GUI

The track with the lowest number will be the master track, whereas the other one will serve as slave track. That means that you should change the values on the first one to modify the values on both tracks, not vice versa, as it wont work [2]

Unlinking

- Open the UI with the toolbarButton that you just set up (or by running the *Marini_ReaLink_Ui_Toggle.lua*)
- Select on the UI the pair you want to remove
- Hit the *"-"* button on the script GUI

Note that this script links fx parameters, not the track values themselves (volume, pan). That can be done without any extension

4 How it works

The linking algorithm basically links the first instances of the same plugin, without considering the order in which those are put. Suppose we have

$$\text{Track1: } \{A_1, A_2, A_3, B_1, C, D_1\}$$
$$\text{Track2: } \{A_1, A_2, B_1, D_1, D_2, E_1\}$$

Then the plugin pairs to be linked would be

$$\{\text{Track1}_{A_1}, \text{Track2}_{A_1}\}$$
$$\{\text{Track1}_{A_2}, \text{Track2}_{A_2}\}$$
$$\{\text{Track1}_{B_1}, \text{Track2}_{B_1}\}$$
$$\{\text{Track1}_{D_1}, \text{Track2}_{D_1}\}$$

so the following fx remain unlinked:

$$\text{Track1}_{A_3}, \quad \text{Track1}_{C_1}$$
$$\text{Track2}_{D_2}, \quad \text{Track2}_{E_1}$$

Long story short, that means that if the 2 fx chains are the same, then every fx will be linked. If not, only common FX will be linked

[2] If you modify enything on the slave track it will just return to the position in which it was. The link is not bi direcional to make the linking algorithm more robust