Unofficial FAQ & Workaround for u-he Plugins on Linux

Technical details about the plugin implementation

- At u-he we have an internal framework which provides a portable scene graph (Win, Mac, Linux/X11), that is how we render the UI
- On Linux a big issue is that there is no standard UI toolkit which is fit for plugin development:
    - Both Qt and Gtk are designed to be used in a “main application”: they use singleton and one might not mix qt4 with qt5 and gtk2 with gtk3. So it is unsafe to use those toolkit to make plugin UI as the plugin UI might runs in the same process as the host or other plugin UI.
- While the u-he framework is advanced, it is not designed to have multiple windows, proper text input (some host just grab all the keyboard input). U-he is using the OS widgets on Windows and Mac to display dialogs, input widgets and contextual menus.
    - On Linux we use gtk3 for those standard dialogs, yet we can’t use them in the current process because of the previous points. So we create a child process to display the menu and our dialogs.
        - There is no performance issue because it is run on the UI thread and we only spawn those process when the user is triggering them

Hosts & Workaround

Bitwig Studio

- Works great

Renoise

- Once we had an issue, because on Unix, when you create a process (fork(), vfork()) you have to wait for your child (waitpid()), and Renoise did call wait() when it receives SIGCHLD (a child process finishes) and so wait for the child process, while the resource belong to the u-he plugin and the u-he plugin was waiting also for the child.
    - To sum up, Renoise was garbage collecting resources even if it did not “own” it.
    - I don’t know if they fixed it today.
- Otherwise works fine

Ardour

- Works great (v4.7 tested)

QTractor

- TODO

Carla

- Works great

Mixbus

- Works great
- Some users reports that the dialogs do not show properly

Tracktion

- TODO

Other issues

- Scrolling is not always consistent
    - Especially in the preset browser