Playlist Tools

(aka Playlist-Tools-SMP)

https://github.com/regorxxx/Playlist-Tools-SMP

regorxxx

January 18, 2023

Contents

Part I

Introduction

1 Why do we need more tools with so many components already available?

1.1 Offline tools in a forced online world

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.2 MusicIp and LikeADJ sucessor

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.3 Open source and user configurable data

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

1.4 Full fledged HTTP controller integration

There are multiple Foobar2000 controllers out there, like foo_httpcontrol. Obviously, they are limited to the features integrated on the component so many advanced tools are out of question. SMP integration could bypass that limitation since it essentially allows to work around almost any Foobar2000 feature in multiple contexts (playlists, library, etc.), even the file system.

Regrettably, SMP is not integrated into any of those HTTP controllers... but that can be remediated with the use of 'ajquery-xxx' controller. Since the component allows to call arbitrary main menu entries, the 9 hardcoded menus by SMP can be called with the online controller when using the appropriate template.

The interesting part comes when using Playlist Tools along the controller, since the scripts can dynamically set those menus, provide names and descriptions, icons, allow output device and DSP selection, etc. Essentially full server control or running any of these tools from any device.

1.5 Expand Foobar2000 queries: macros and configurable menus

Dynamic Queries: queries which adapt to the currently selected track. i.e. placeholders tags are substituted with the actual values of the currently selected track, then the query is evaluated as usual. Queries created this way are pretty situational, save a lot of writing time and are meant to be used by multiple playlist creation tools. Pools: playlist creation similar to Random Pools component. Multiple playlists \library sources (pools) can be set to fill a destination playlist.

Configurable selection length per source, query filtering, picking method (random, from start, from end) and final sorting of destination playlist. They may even use dynamic queries changing the way the pools behave according to selection (for ex. a pool which outputs tracks with same key than selected track + another one which outputs same genre tracks), the main limitation of Random Pools component.

1.6 Expand Spider Monkey Panel capabilities: macros and configurable menus

Fully configurable submenu entries: shift + left click on menu button allows to switch tools functionality. Individual tools or entire submenus may be disabled/enabled. When all entries from a tool are disabled, the entire script files associated are omitted at loading.

Macros: allows to record and save the menus entries used, as a macro, to be called later. Automatic custom playlist creation and edits without limits. Works with all tools. (only limitation are popups, which still require user input)

User configurable presets: many tools allow you to add your own presets (for ex. Standard Queries) as menu entries for later use. They may be used along macros to greatly expand their functionality, exported and imported as "addons". Global shortcuts (experimental): global shortcuts without requiring panel to be in focus associated to some tools. Shown on the related menu entries tabbed to the right. Experimental feature, read the popup before activating it.

Include other scripts (experimental): easily include ('merge') multiple SMP scripts into the

same panel, thus not wasting multiple panels. Useful for those scripts that don't require any UI, user interaction,... like scripts which set the main menu SPM entries (File\Spider Monkey Panel).

1.7 FLS slots limit (DLL's): Reduce components loaded with better replacements

One of the main limitations of windows (and thus Foobar2000) is the maximum number of plugins (DLL's) that can be associated to a given process. Thus, in some installations, specially those using VSTs, when the limit is reached strange things start happening: random crashes, bugs, etc. Something I have experienced myself while running or installing a few VSTs as DSPs within the software.

It's not so hard to reach that limit since many components use multiple DLL's! When you count the ones by foobar itself, VSTs, etc. as soon as you configure a bit your installation you come into problems. Therefore, Playlist Tools is a solution that can help in that sense, replacing multiple components whose functionality is not only included but also improved: Random Pools, Playlist Revive, Best version picker, Database Search, ...

Part II

Installation

2 Scripts files and dependencies

Spider Monkey Panel 1.5.2 or higher is required. Only stable releases are supported.

Copy all files from the zip into 'YOUR_FOOBAR_PROFILE_PATH\scripts\SMP\xxx-scripts: **If the folders don't exist, create them. Any other path WILL NOT work without editing the scripts**. (see '_TIPS and INSTALLATION_vX.X.jpg'). Multiple scripts may share some files (specially helpers) so overwrite if asked to do so. Then load the script named **'buttons_toolbars'** into a SMP panel within foobar [4].

- For **standard installations**:
  'C:\Users\yourUser\AppData\Roaming\foobar2000\scripts\SMP\xxx-scripts\...'
- For **portable installations >= 1.6**: '.\foobar2000\profile\scripts\SMP\xxx-scripts\...'
- For **portable installations <= 1.5**: '.\foobar2000\scripts\SMP\xxx-scripts:.'

If you upgraded to >1.6 from an older portable version then it may be possible that the 'profile' folder does not exist. In such case you have to create it and move all the configuration folders/files to it, where they should reside (instead of the root of foobar2000 installation path). If you don't move all the configuration folders/files, then, on startup, default values will be used for things not found, probably "losing" the theme or other customization. You may "fix" it later moving the missing files which still reside in the root. May take some tries to do them all.

- **Some native folders and files which must be moved include:** index-data, js_data, component-updates, configuration, crash reports, user-components, foo_spider_monkey_panel, library, playlists, theme.fth, LargeFieldsConfig.txt, version.txt
- **Some extra folders from other components which must be moved include (non extensive list):** autobackup, dvda_metabase, foo_httpcontrol_data, foo_youtube, images, lastfm, python, sacd_metabase, vst-presets, yttm, minibar.db, playlist-tree-0.pts, playlist-tree-1.pts

Additionally, some fonts are required on the system. They can installed by double clicking on the '.ttf' files and then selecting the 'install' option (requires Admin rights):

- **Font Awesome**: found at '._resources\fontawesome-webfont.ttf'

3 Portable installations tip

When the script finds it's being loaded within a portable installation, it will set the default paths using relative paths. It will also warn with popups and/or the console about the -non recommended- use of absolute paths on portable installations. For more info see [**??**].

4 Installation within Foobar2000's UI

Once all files have been installed to the right paths, along their dependencies, an Spider Monkey Panel must be added to the current layout anywhere on the UI. There are some minor differences between the Default UI (DUI) and Columns UI (CUI), but in both cases the layout can be edited using the menu **'View\Layout\Live editing'**:

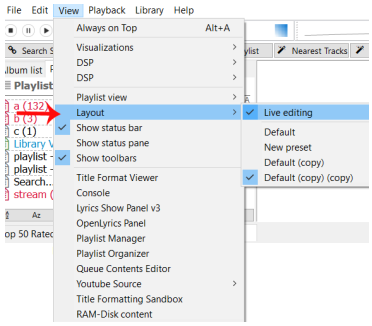


Figure 1: Editing the layout (CUI).

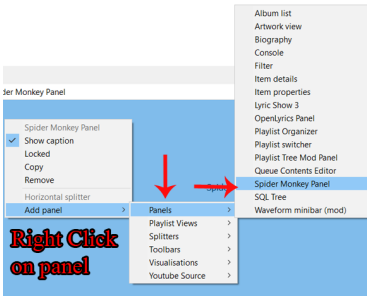


Figure 2: Adding a new SMP panel.

An SMP panel is only a blank UI panel with no further functionality until a '.js' script file is loaded. Multiple SMP panels can be added to a layout and they may point to the same script files without any problem; every panel is considered an independent entity. Multiple Playlist Manager panels may be added to the UI this way [**??**], either to have different views of the same playlists or different folders tracked. After the panel has been added to the current UI, it must be configured to load the main '.js' script file. It can be done by right clicking on the blank SMP panel:

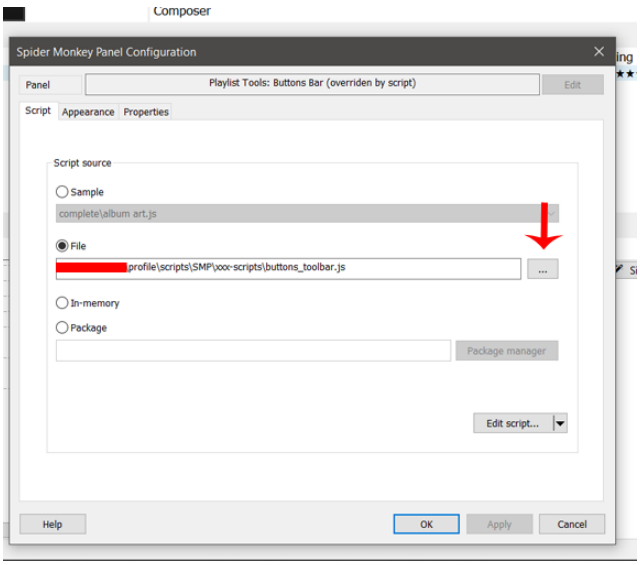


Figure 3: Loading the **'buttons_toolbar.js'** script within a SMP panel.

Once the script has been properly installed, buttons may be added [12] and presets loaded

[11].

5 Installation as a toolbar panel *-requires CUI-*

This installation variant requires Columns UI to be used. Foobar2000 allows to display at top an indefinite number of toolbars (rows), and it's possible to integrate SMP panels into them using CUI.

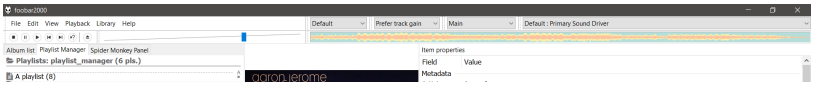


Figure 4: A pretty standard installation with 8 toolbar panels (see dotted lines) in 2 rows.

To do so, right click on the main menus toolbar (the blank space or even the menus), and click on 'Panels\Spider Monkey Panel'. It also works clicking on the dotted line at left of every row (to insert it on specific rows).

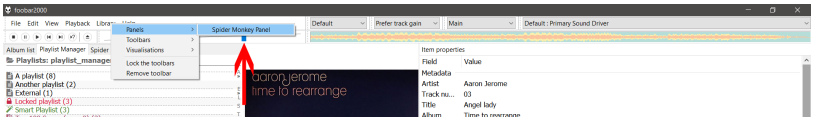


Figure 5: Toolbar configuration menu at right click.

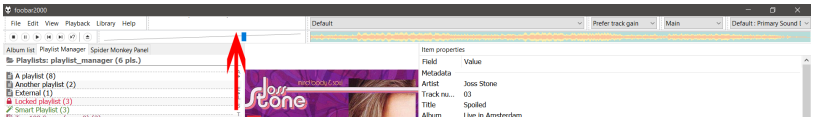


Figure 6: A new toolbar SMP panel will appear, with a white background.

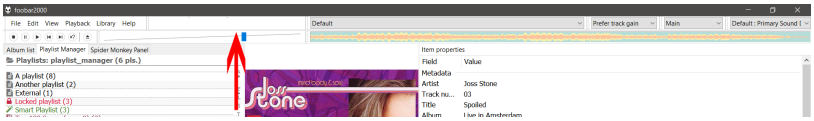


Figure 7: In case Windows uses a colored theme, the panel color mismatch will be obvious.

At this point, follow the default installation steps [4] seen before, by right clicking on the new panel.

Clicking again on the same menu entry will remove the panel but **there is a trick to create more panels: Press shift while clicking** on 'Panels\Spider Monkey Panel'. It will append new toolbar panels to the existing ones. Pressing without shift, when there are multiple panels created this way, will remove the first one available starting from the current row.

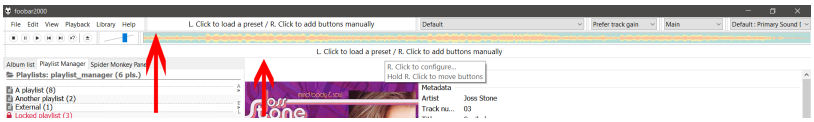


Figure 8: 2 SMP buttons toolbar panels already installed.

As any native toolbar, they may be moved, configured or resized clicking on the dotted line at their left.

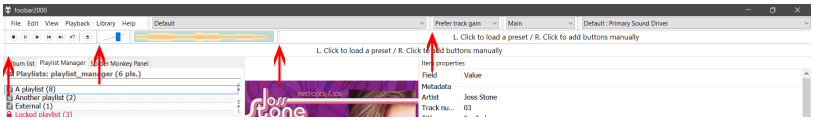


Figure 9:

It may be wise to set the edge style to 'None' and check 'Use pseudo-transparency' at the SMP configuration panel\Appearance, so the panel is integrated following the current global theme (instead of the white background seen at top).

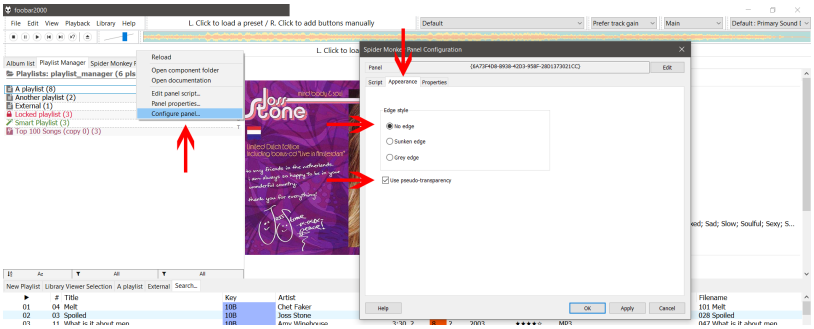


Figure 10: Configuring appearance.

Following all these steps, SMP buttons panels work exactly the same than native buttons. This installation variant has also multiple advantages:

- Theme is managed by native Foobar2000.
- Size is automatically adjusted and standardized between all toolbars.
- There is no need to rearrange the current layout for new panels, it works in any layout as an ad-hoc toolbar.
- Panel(s) can be easily moved or resized as the default toolbars without affecting the layout.
- It's trivial to create multiple toolbar panels in the same row. This could be done for performance reasons when using resource intensive-tools (since one button may be blocked "processing" but the other panels are independent).
- While the same results may be achieved using the layout installation method, it usually requires a lot more of work or thought to integrate it properly on complex themes.

6 Installation along Buttons framework compatible scripts

When using this script or other button tools, you may want to use the tools within an existing toolbar panel (instead of creating another independent panel). To do so skip loading the button file

within a new panel and go to the toolbar section [III] for further instructions. A non comprehensive list of compatible button-scripts includes:

- Fingerprint-Tools-SMP
- Device-Priority-SMP
- ListenBrainz-SMP
- Search-by-Distance-SMP
- ...
- For the full list look here.

7 Wine installations

Script is 100% compatible with unix systems although some points should be taken into consideration:

- 'Segoe UI' must be present, as it is the font by default on the script's UI.
- In case the font is not found (check console log), revise your default font on Wine. Changing it to a different one than 'Segoe UI' may give problems or crashes in some cases.
- [Additional] required fonts must be installed to avoid crashes. Is not only eyecandy.
- At some point the script may use external CMD tools which must be compatible with 32 bit systems in Wine (since that's the most popular version, with IE). In this case 32 bit binaries for 7z and CmdUtils are bundled.
- Script may throw an error after installation (either a foobar crash or panel crash) due to 'missing files'. There is a fix for that, check [**??**] and [**??**]. It's a SMP bug, don't report it.
- Wine installations are specially sensible to missing files, wrong paths, etc. which may lead to crashes or bugs usually not found on Windows installations. Have that in mind and double check your installation procedure before reporting an error [**??**].
- All scripts try to use wine-friendly methods, focusing on configuration settings that can be changed via menus or the UI panel, instead of using HTML (which only works on Windows), known working fonts, etc.
- Please read SMP Wine page and feel free to report (me) any additional problem with these scripts.
- Read Wine foobar thread for more info and tips.

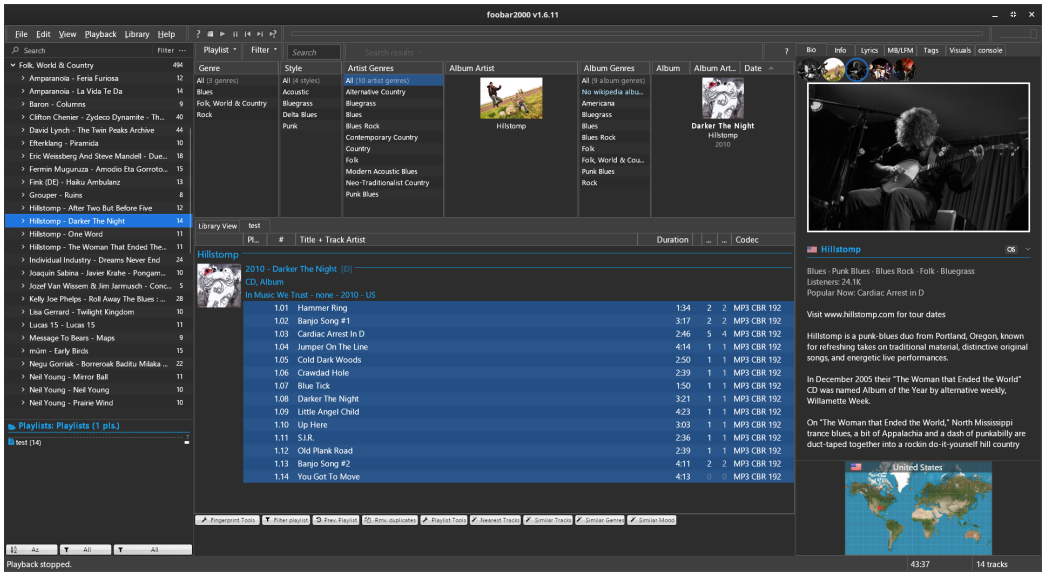


Figure 11: Script working on Wine along Playlist Tools and World Map.

8 Updating from older script versions

There is nothing special to do but following the standard installation steps and overwriting all [old] files as needed. In case an update changes something relevant or requires an additional step, that will be warned on the release itself.

9 Nightly versions

it's possible to download the more recent version directly from the repository, instead of the releases page.

Figure 12: Releases vs WIP.

In such case it's possible that some changes are not fully tested; the list of "unreleased" changes is always shown -and updated- at the changelog. The installation instructions are the same than the other cases. To download the current repository, look below:

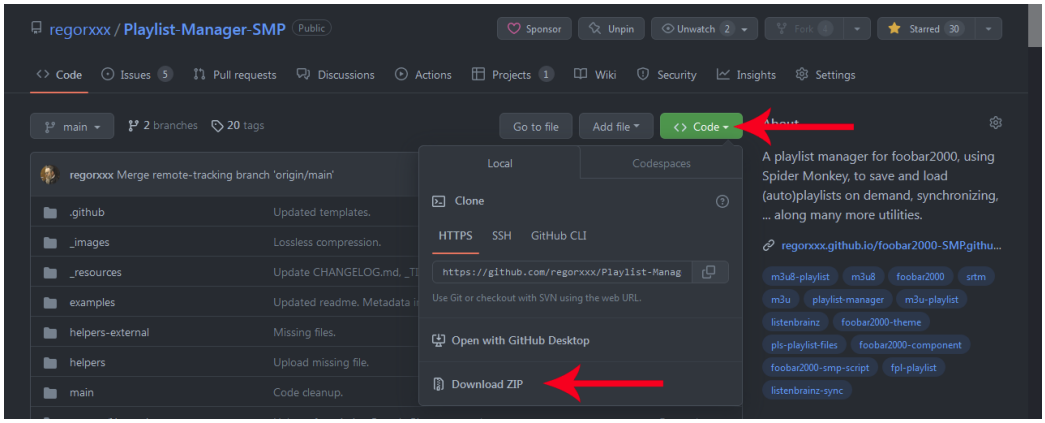


Figure 13: Downloading most recent files on repository as zip.

Part III

Toolbar framework

Part IV

UI

15 Features

- Pre-defined or custom buttons may be added, moved or deleted.
- Tooltip for every button.
- Colors for button text and toolbar (configurable).

15.1 Tooltip

Tooltips show different info according to the mouse position:

- xxx:

16 Customization

16.1 Custom color

- Background: toolbar background.
- Buttons text: for text within buttons, in case you change the SO theme.

Part V

Other scripts integration

This is a non comprehensive list of other Spider Monkey Scripts or plugins which may be used along the manager or whose features are designed to work together:

- Playlist-Manager-SMP:
  * **Random Pools:** Pools may use tracks from playlists files tracked by the manager, not requiring to have the playlists loaded within foobar. i.e. Random Pools component-like playlist creation, using not only queries as sources, but also other playlists or playlists files. See [18].
  * **Playlist Revive:** Finds and replaces dead items on loaded playlists or selection. Meant to be used along dead items checks on playlist files [**??**]. First check all playlist files, then load those with dead items and use Playlist Revive.
  * **Duplicates and tag filtering:** The manager allows to report playlists with duplicated items, but it's limited to entries with same path. This tool expands Foobar2000 native functionality of removing duplicates, allowing to find duplicates by tags (for ex. any track with same Title - Artist).
  * **Import track list:** Takes a plain text list of tracks (for ex. Title - Artist) and finds matches on library to create a playlist. Meant to be used for playlist importing when the track list does not follow an standard format or there are no paths provided[1]. Instead of sharing a list of files, list of tracks may be used which work universally no matter your configuration. Non found items are simply discarded.

[1] Technically that is not a playlist. But note playlists with relative paths may easily be considered a track list as long as you discard the '.\' part. In other words, a plain-text list can be retrieved from playlists in many cases.

Part VI

FAQ

- **What's json?** It's a standard file structure. Check https://en.wikipedia.org/wiki/JSON for more info.
- **What's asynchronous execution?** Execution of some code done on the background (usually on iterative steps), thus not blocking the UI on the process. For ex. external playlist files loading on native Foobar2000
- **Some lines in imported text files are not displayed/recognized properly:** That probably points to code-page detection errors by SMP on reading. UTF-8 files without BOM may be incorrectly identified and thus not parsed as intended, consider using UTF-8 with BOM before importing track lists. Multiple code checks have been added to minimize these situations in any case.
- **The tools don't work as expected at random instances:** Check the installation process and paths. Take a look at the file named '*_TIPS and INSTALLATION.txt*' which should be found along this readme.
- **I have found a possible bug not related to a wrong installation:** please, create a new issue at github: https://github.com/regorxxx/Playlist-Tools-SMP.

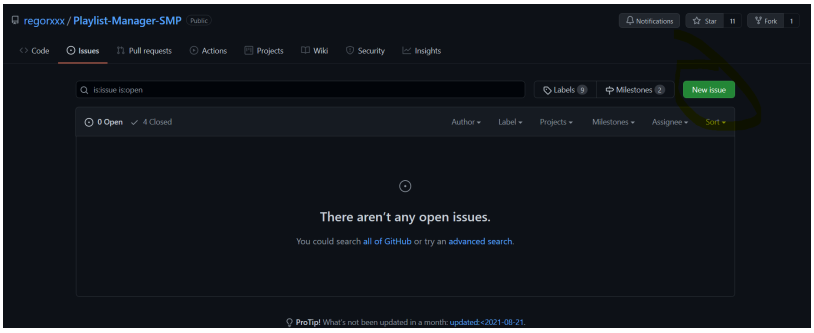


Figure 14: Opening a new issue at github.

Part VII

Advanced tips

17 Example

- Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

18 Pools using Playlist-Manager-SMP

- **Feature is enabled automatically when using both scripts in the same foobar2000 instance**. As previously noted, Playlist-Tools-SMP [V] has a random pools feature which can use playlists as sources to output a random list of tracks from all its pools. Playlist-Tools-SMP checks periodically if there is a Playlist Manager panel and retrieves the list of playlist being tracked to use them as source if required. In other words: there is no difference between loaded playlists within the UI or playlist files.

- There is a **preset example at "presets\Playlist Tools\pools\test_playlisManager.json"** which uses a playlist named "test" as source (after importing it). The file may be used as reference to create your own presets.

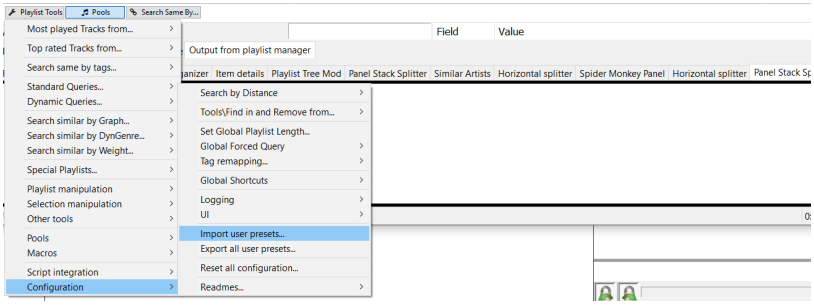


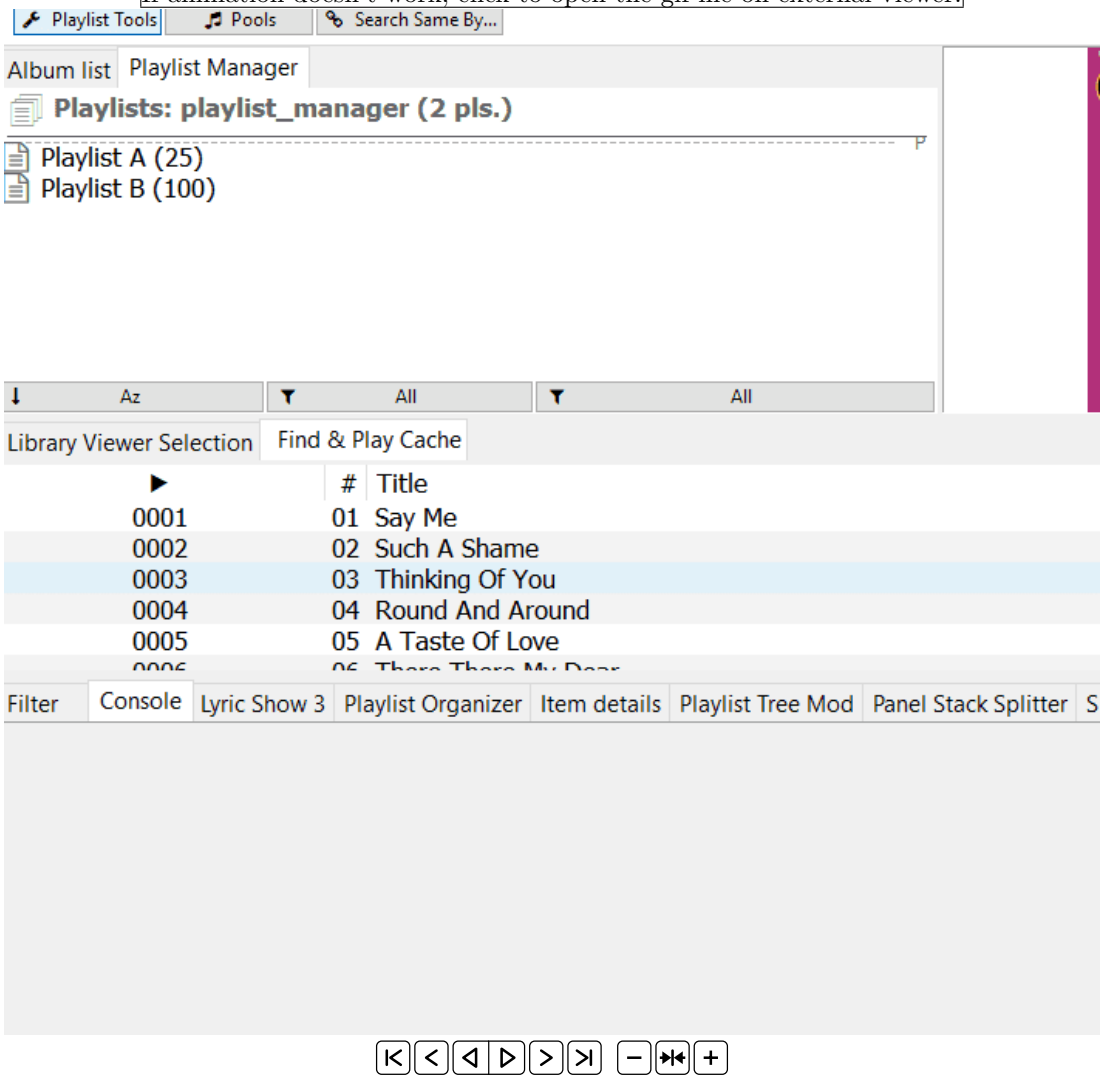
Figure 15: Importing a preset file in Playlist-Tools-SMP.

- For example **to use "Playlist A" and "Playlist B" as sources, a preset file would have to be set like this**:

```
{
    "readme": "Playlist A and Playlist B as sources, 25 tracks per playlist.",
    "pools": [{
        "name": "Playlist Manager test V2",
        "pool": {
            "fromPls": {
                "Playlist A": 25,
                "Playlist B": 25
            },
            "query": {
                "Playlist A": "ALL",
                "Playlist B": "ALL"
            },
            "toPls": "Output playlist",
            "sort": "%playlist_index%",
            "pickMethod": {
                "Playlist A": "random",
                "Playlist B": "random"
            }
        }
    }]
}
```

- Alternatively, a **custom pool may be directly executed once via menus -or added as a new entry for later use-**:

If animation doesn't work, click to open the gif file on external viewer.



- **Playlists tracked by the manager(s) may be used along other sources** like playlists within the UI, library (+query) and other complex functions:

```
[...]
"fromPls": {
    "Playlist A": 25,
    "_LIBRARY_0": 7,
    "_SEARCHBYGRAPH_0": 13
},
```

- Playlist follow this rule when **trying to find a match: loaded playlist first, then matching playlists by name (metadata) and finally by filename**.

Part VIII

Technical notes

- Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Part IX

Known problems

- **Spider Monkey Panel 1.5.2 may crash -without known reason- at startup when trying to install the manager. The error will warn about script files not being found, although they are in place.** This is due to incorrect relative path handling on some systems. In such case, **take a look at the '_switchPaths.zip', decompress it and follow its instructions**. Scripts will be automatically edited to use absolute paths at script loading, thus fixing the problem. The cmd file (.bat) will have to be rerun on future playlist manager updates. Obviously this is only a workaround and a proper fix is expected on posterior SMP releases.

- **Spider Monkey Panel (any version) may incorrectly identify code-page while reading text files**, thus reporting wrong values at some instances. This usually happens while reading playlist files or importing tracks lists, with track names or artists using exotic chars (ä,â, ...) getting corrupted. Multiple code checks have been added to minimize these situations in any case. See FAQ [VI].