PSR

Registration Shuffler

# Design: Redesign of version 1.0

Version 1.0

December 28, 2011

Dennis Schulmeister

dennis@developer-showcase.de

# Contents

# 1 General specifications

## 1.1 Purpose of the redesign

Over the past three years the PSR Registration Shuffler evolved from a small tool to a medium-sized application with support for different arrangers and many useful features. Therefor the code size has already grown to 6200 lines. In the meantime some problems have become clear which led to the decision to not maintain the old code base anymore:

Unexperienced users didn't understand the basic principle of importing registrations. Instead they expected to be able to directly edit files which were previously created on their instruments. Also they didn't understand the difference between registrations and registration banks. Both points and the fact that there has been no real manual made users perceive the application as hard to understand and therefor hard to use.

Some of the technologies upon which the old version was build is soon to be considered obsolete. Python 2.5 is followed by Python 2.7 or 3.2. `GTK+` 2.12 is a very old version of the 2.x branch. The recent version is `GTK+` 3.0 or 3.2. Both versions use new introspection based python bindings and are currently adopted by major Linux distributions. And finally there's a dependency on the `kiwi` library which is not maintained anymore.

Besides that the code itself has some weak points which need fixing: Names of functions, classes and methods follow the java convention instead of `PEP 8` and decorators don't use the abbreviated `@`-syntax. Some magic `CPython` voodoo is used to implicitly detect subclasses for registrations and bank files. Also package and module names are not necessarily unique. And finally the class design is broken in some ways regarding static methods and scanning the data directory for supported files.

Regarding the small code size fixing all this issues is too expensive and could be considered a major rewrite. For that reason it's much easier to start from scratch with a cleaner design and more recent technologies.

## 1.2 Related documents

More information can be found in the following documents:

`ds2-file_formats.pdf`
Technical description of all used file formats. This includes internal files (configuration, data files) as well as data files by supported instruments.

## 1.3 Used technologies

Easy creation of cross-platform distributions and small installation size are considered major goals for the new version. Therefor the new version shall not rely on Python and `GTK+` anymore. Instead the following technologies shall be used and dependencies on other libraries must be carefully considered with hindsight to cross-platform packaging (see next section).

- `C++` 2011
- `wxwidgets` $\geq$ 2.8
- `premake4` $\geq$ 4.3

For development a standard GNU/Linux environment is assumed and some utility programs might not be readily available for other platforms. However through the use of `premake4` it should be fairly easy to at least compile the application on other operating systems.

Documentation and design documents may use Texlive 2009 (or newer) and the KomaScript classes. Makefiles are used for simple scripting tasks.

Mercurial is the preferred source control software and all sources are hosted on Bitbucket:
`https://bitbucket.org/DennisSchulmeister/psr_reg_shuffle1`

## 1.4 Supported platforms

Development focuses on a Debian GNU/Linux platform. Therfore all development tools and the application itself must be easily executable in that environment. However each release must also be able to run on Microsoft Windows. The minimum required Windows version shall be Windows XP.

| Platform | Distribution format |
|---|---|
| GNU/Linux, Unix | Source tarball |
| Microsoft Windows | Binary installer |
| (Apple Mac OS X) | Application bundle |
| Others | Source tarball |

A Macintosh version shall be distributed in form of a self-contained application bundle. This is considered a secondary target, though. Currently more research on cross-building Macintosh applications on Linux is needed.

If possible, all binary distributions shall be cross-built within a pure GNU/Linux environment.

## 1.5 License terms

Just like the previous version the application shall be licensed under the terms of the GNU General Public License version 3 or later. This applies to the program, documentation and all provided example files. Only libraries which are compatible to the terms of the GPL may be used in order to write the program.

Besides that the toolchain needed to build the program and all related files must be free software which is readily available to developers. Therefor all tools used in the build process must at least conform to the BSD license terms.

2

# Functional specifications

## 2.1 Feature description

Though the new user interface is very different from the old version all core features of the old version shall remain available. This means it shall be possible to …

- Create new registration banks
- Import registrations and split them into individual files
- Edit registration banks without importing them
- Sort and shuffle registrations
- Printing setlists from existing banks
- Export lists in different file formats

Additionally an online help function shall be available. However the old page which describes all supported keyboard models isn't needed. Later versions might allow to display and change the contents of individual registrations.

The main purpose of the application is to assemble bank files from already existing registrations. For this to work it has been necessary to import all available registrations into a local directory where the program created individual files for each registrations. Of course all files created with the old version must remain usable. However new files created by the new version shall use an updated file format which may contain metadata. Also it shall be possible to skip the import step at all. Instead when the user selects the data directory the program needs to scan all files within that directory for available registrations. This includes individual registration files as well as bank files.

Another characteristic feature is to reorder and edit registrations of one ore more bank files without importing them first. A similar function called *Quick Edit* has already been available in the old version. However it has only been possible to edit the contents of one bank file at a time. There has been no way to open many bank files at once in order to move registrations between banks. Starting with the new version this shall be possible for all arrangers which store registration banks to individual files. For older arrangers which store all banks into one large file it is sufficient to open one file only but it must still be possible to move registrations between banks. Also it should be possible to rename banks and registrations and to delete registrations.

When a registration is deleted the user may choose whether it's replaced with a special empty registration or whether all following registrations move up one place.

The sort function shall allow to sort all registrations within the edit buffer. For the first incarnation it shall be possible to sort registrations by name ascending or descending. Also it shall be possible to shuffle them in random order. Later versions might add sorting by style or tempo. The old option to keep registrations with similar names together. The user may specify that registrations belong together if their names differ only in a certain number of trailing letters. Besides that another option shall exist that specifies that any subsequent registrations whose names don't begin with alpha-numeric characters (e.g. space or dash) may not be separated from their respective precursor. Consider the following example of registrations of which some must be kept together:

| | |
|---|---|
| **1** | C'mon everybody |
| **2** | Fly, fly away |
| **3** | New York, New York (1) |
| **4** | New York, New York (2) |
| **5** | New York, New York (3) |
| **6** | Romeo and Juliet |
| **7** | - Sax Solo |
| **8** | - Ending Guitar |

Independent from the sort function an option to automatically name banks is needed. The most simple way to calculate bank names would be a simple two-digit numeration like *01*, *02*, . . . or *Bank 01*, *Bank 02*, . . . In may cases however this might not be good enough. In that cases the first letters of the first and the last registration of each bank shall be used to calculate a short label like *Like A-Lonely*.

Through the export and the print functions it shall be possible to create lists of all registrations within the edit buffer. The user must be able to choose which data to export (e.g. registration name, arranger model, tempo, style name). Similar to the old version at least the following file formats shall be exported:

- Plain text with no formatting
- Plain text with simple formatting
- Structured `CSV` files

Printed lists shall have a simple tabular layout which visually separates the banks from each other.

## 2.2 Main window layout

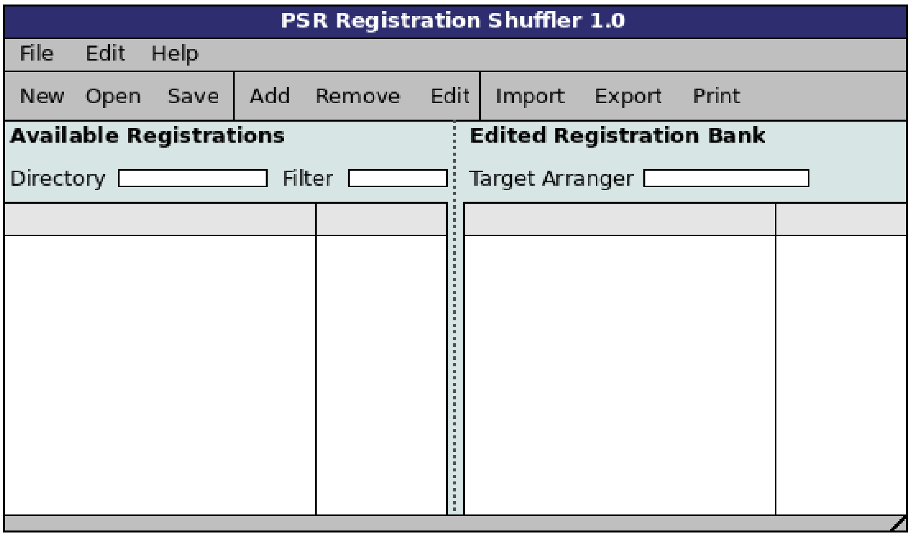


Figure 2.1: Sketch of the main window

One major concern of the old application has been, that users were not always able to easily find out how certain things are done. For one thing the idea of different tab pages for different tasks was surprising. Also within the *Create Bank Files* page the relationship between the dropdown boxes *Data Directory*, *Keyboard Model* and *Filter* hasn't been obvious. For that reason the new user-interface follows a more traditional vertical layout with a menubar, toolbar, main area and status bar.

The menubar shall contain three menus called *File*, *Edit* and *Help*. Those menus shall contain at least the following entries. Whenever possible they must match the functions on the toolbar.

| **File** | **Edit** | **Help** |
|---|---|---|
| New | Add Registrations | Help |
| Open | Remove Registrations | Website |
| Save | Insert Empty Registration | - - - - - - |
| Save As | - - - - - - | About |
| - - - - - - | Edit Registration | |
| Import Registrations | Shuffle Registrations | |
| Export Registration List | Name Banks | |
| - - - - - - | | |
| Page Setup | | |
| Print | | |
| - - - - - - | | |
| Quit | | |

The following functions should be available on the toolbar: *New*, *Open*, *Save*, *Add Registrations*, *Remove Registrations*, *Edit Registration*, *Import Registrations*, *Export Registration List* and *Print*.

The largest portion of the window is covered by the main area which sits between the toolbar and the statusbar. It is vertically divided into two equal-sized parts which are called *Available Registrations* and *Edited Registration Bank*. The user may move the boundary between both parts. On the left side are two dropdown boxes called *Source Directory* and *Filter*. On the right side is only one dropdown widget called *Target Arranger*. All dropdown boxes are aligned to one line. Below them are two lists which show available registrations on the left side and the edit buffer on the right side.

Once the user has selected a *Source Directory* the program scans that directory for any available registrations. This search includes individual `*.regfile` files created by the import function as well as usual bank files. All found registrations are shown in the list below. If the user has selected a *Target Arranger* she is able to move registrations to the right list in order to assemble new banks. Also it is possible to choose between the following filter options. Each option restricts the list of available registrations accordingly:

- All Registrations
- Compatible Only
- Target Arranger Only

If *All Registrations* is selected, all registrations are shown no matter of the arranger model on which they were created. *Compatible Only* means to show only registrations which can be added to the right side. This may involve registrations which were created on different models. If only registrations should be displayed which were created on the target arranger the option *Target Arranger Only* has to be used.

On the right side the edit buffer can be seen. It is best displayed by a hierarchical list which groups registrations by banks. The idea here is that the user adds or removes registrations to that list but cannot create new banks explicitly. Instead banks are created or removed implicitly as required. However it must be possible to rename registrations as well as banks. For easy reference the displayed bank names shall be prefixed with a serial number. Also all registrations within a bank shall be numbered. Registrations may be added or removed either by drag'n'drop or dedicated commands in the toolbar and edit menu.

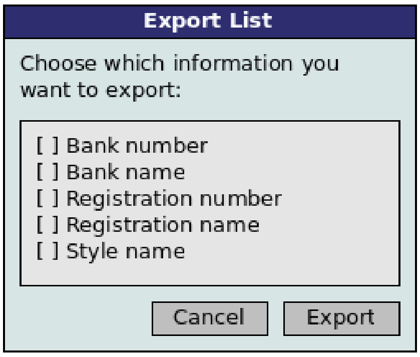
On the left side it shall also be possible to rename registrations. If done so the underlying bank or registrations files need to be updated silently. In order to allow insertion of empty registrations the left list must contain a special entry called *Empty Registration*. Regardless of the sort order this entry is always the first item and it is displayed different from the other registrations.

## 2.3 File functions

*New*, *Open*, *Save*, *Save As* work as expected. They affect the edit buffer which is shown on the right side. Since newer arrangers store each bank to an individual file it must be possible to open many files at once, provided the file formats are compatible. Depending on the selected arranger model the *Save* and *Save As* windows must be file or directory selectors. If the selected arranger stores single files for each bank it shall only be possible to select a destination directory where the bank files are created. Otherwise a file selector which suggests the most common filename needs to be used.

*Import Registrations* must be inactive unless registrations have been opened with the open function. Any other of the aforementioned file functions leads to the import function being disabled. This function reads all registrations from the edit buffer list and stores them into individual `*.regfile` files in the *Source Directory*. All imported registrations immediately appear inside the *Available Registrations* list.

*Export Registration List* allows to create printable lists of all registrations within the edit buffer. According to the feature description several file formats like `*.csv` or plain text should be supported. Before export the user must choose which information she wants to export. This may include registration values as well as meta-data.



Export List

Choose which information you want to export:

[ ] Bank number
[ ] Bank name
[ ] Registration number
[ ] Registration name
[ ] Style name

Figure 2.2: Select exported data

*Page Setup* opens the native printer configuration window. Here the user can set options like paper size or duplex printing.

*Print* first opens a similar window like the export window. Afterwards the native print dialog is opened in order to print a simple tabular list of all registrations inside the edit buffer.

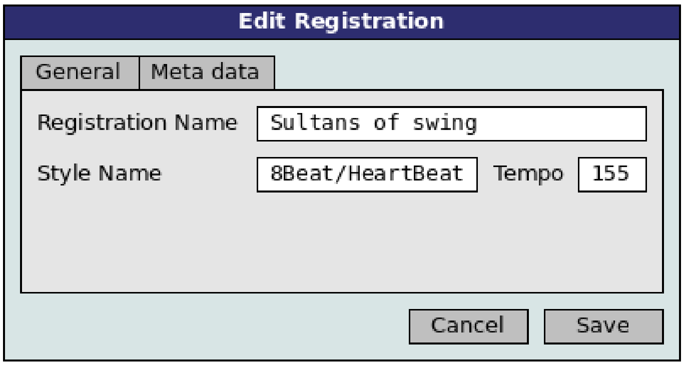
*Quit* checks whether there are unsaved changes and terminates the program.

## 2.4 Edit functions

*Add Registrations* simply appends all registrations selected on the left side to the edit buffer. If needed banks are automatically created. *Insert Empty Registration* appends an empty registration to the edit buffer. Registrations which have been added to the buffer may not disappear from the left side. Instead it must be possible to add the same registrations one or more times.

*Remove Registrations* is just the opposite command. It removes all selected registrations from the edit buffer and makes sure that there are no empty banks left over.

*Edit Registration* brings up a window which shows all details about a single registration. If more or less than one registration is selected an error occurs. Whether the selected registration is searched on the left side or the right side depends on the widget focus. Inside the detail window it must be possible to change the registration name and meta-data. All meta-data is presented as a simple name/value list. If supported additional fields like tempo, style, voices and so on may be edited.



Registration Name: Sultans of swing

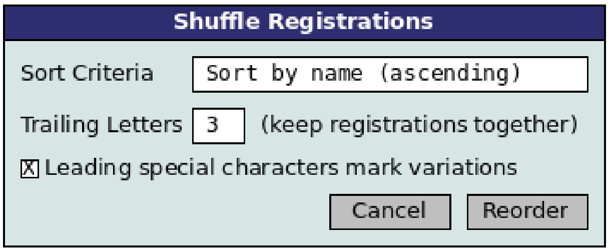
Style Name: 8Beat/HeartBeat

Tempo: 155

Figure 2.3: Select exported data

*Shuffle Registrations* opens a dialog which allows to reorder the registrations inside the edit buffer. This window has three options for the sort criteria and how to keep registrations together which may not be separated (see feature description). Just like the old version the following sort criteria shall be offered:

- Sort by name (ascending)
- Sort by name (descending)
- Random order



Sort Criteria: Sort by name (ascending)

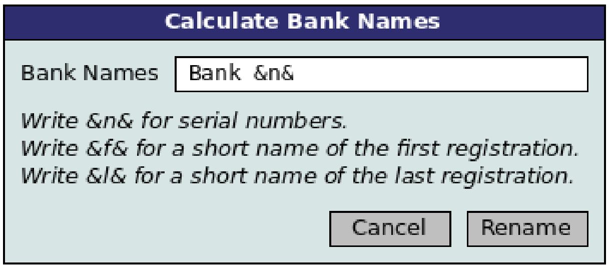
Trailing Letters: 3 (keep registrations together)

☒ Leading special characters mark variations

Figure 2.4: Reorder registrations

*Name Banks* allows to automatically calculate the names of all banks inside the edit buffer. This window has only one entry field for the bank name. Special sequences like

`&n&` need to be replaced with variable values like a serial number. All other characters are just copied to the calculated names.



Calculate Bank Names

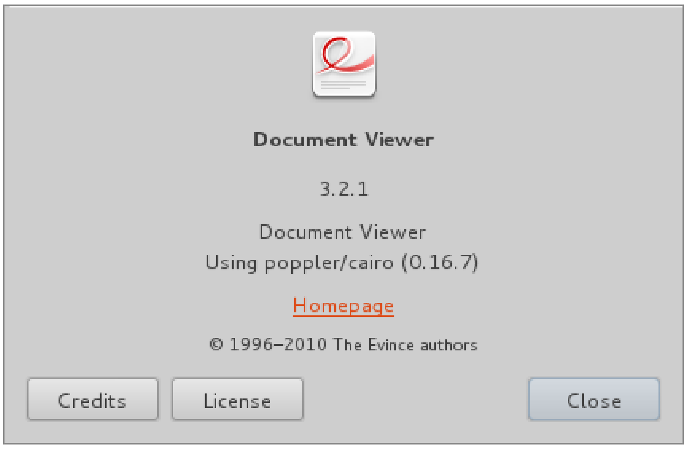
Bank Names: Bank &n&

*Write &n& for serial numbers.*
*Write &f& for a short name of the first registration.*
*Write &l& for a short name of the last registration.*

Figure 2.5: Calculate bank names

## 2.5 Help functions

*Help* opens the included user guide using the preferred `PDF` viewer. *Website* uses the default browser in order to open the address `http://www.psrregshuffle.de`. Finally *About* shows a little window with the program name, version, author and license information.



Document Viewer

3.2.1

Document Viewer
Using poppler/cairo (0.16.7)

Homepage

© 1996–2010 The Evince authors

Figure 2.6: Example about window

# 3

# Included documentation

All distributions shall contain a user guide which is just a PDF file. The manual describes the basic concepts of registrations and bank files and how they relate to the application. The second part goes on to describe the user interface based upon common tasks the user needs to perform.

The Linux and Macintosh distribution shall also contain a manpage. This can be a simple stub which explains how to start the program and how to access the user guide.

Both user guide and manpage shall contain the program name, version number, license information, website and e-mail address. The user guide shall contain an appendix which reprints the full GPLv3 license.