MuseScore Page Settings Changes

Issues

https://musescore.org/en/node/278889

https://musescore.org/en/node/278887

https://musescore.org/en/node/108196

https://musescore.org/en/node/108226

https://musescore.org/en/node/166306

https://musescore.org/en/node/272331

Internationalization and Default Values

The user interface can benefit from several enhancements that coincide with fixing bugs. The UI currently defaults to metric units and A4 page size. Many applications, including MS Office, use the locale settings to choose between at least 2 choices, Metric and US Imperial units.

The QLocale class and the QLocale::MeasurementSystems enumeration provide access to this data point, and MuseScore could use it to set a user's default units to millimeters or inches. That is what MS Office and those other programs do, and it's very user friendly.

But this has implications beyond default page units. MuseScore can also default the preset page size to US Letter vs A4, and the default margins for those two page sizes could be different, thus more standard for the US Letter size.

The other defaulting mechanism that could be enhanced is that page-size drop-down list populated by QPageSize::PageSizeId. It would be user-friendlier to remember the last user selection in that list and restore it at the start of each session.

The page settings default values are in a .mss style file. Should US/inches be a separate file or should values be modified in the code? It affects only a small subset of the styles.

It might also be appropriate to save the user's last page settings values as the default, effectively modifying the default style file. Is there a feature in the Styles dialog to save the current settings as the default? No. Neither is there a way to reset all styles to the default, as with Preferences (a button). Are there any preferences cached as styles? I don't see any. The only overlap is the default style file setting.

All of the above requires 1 or 2 read-only default style files, plus one writable user default file, all standard, without the user setting the default style file. You should still be able to set the user default style file, but it should default to an existing, writable file.

Dialog Box mscore/prefsdialog.ui

There are two standard desktop publishing methods for storing the page units:

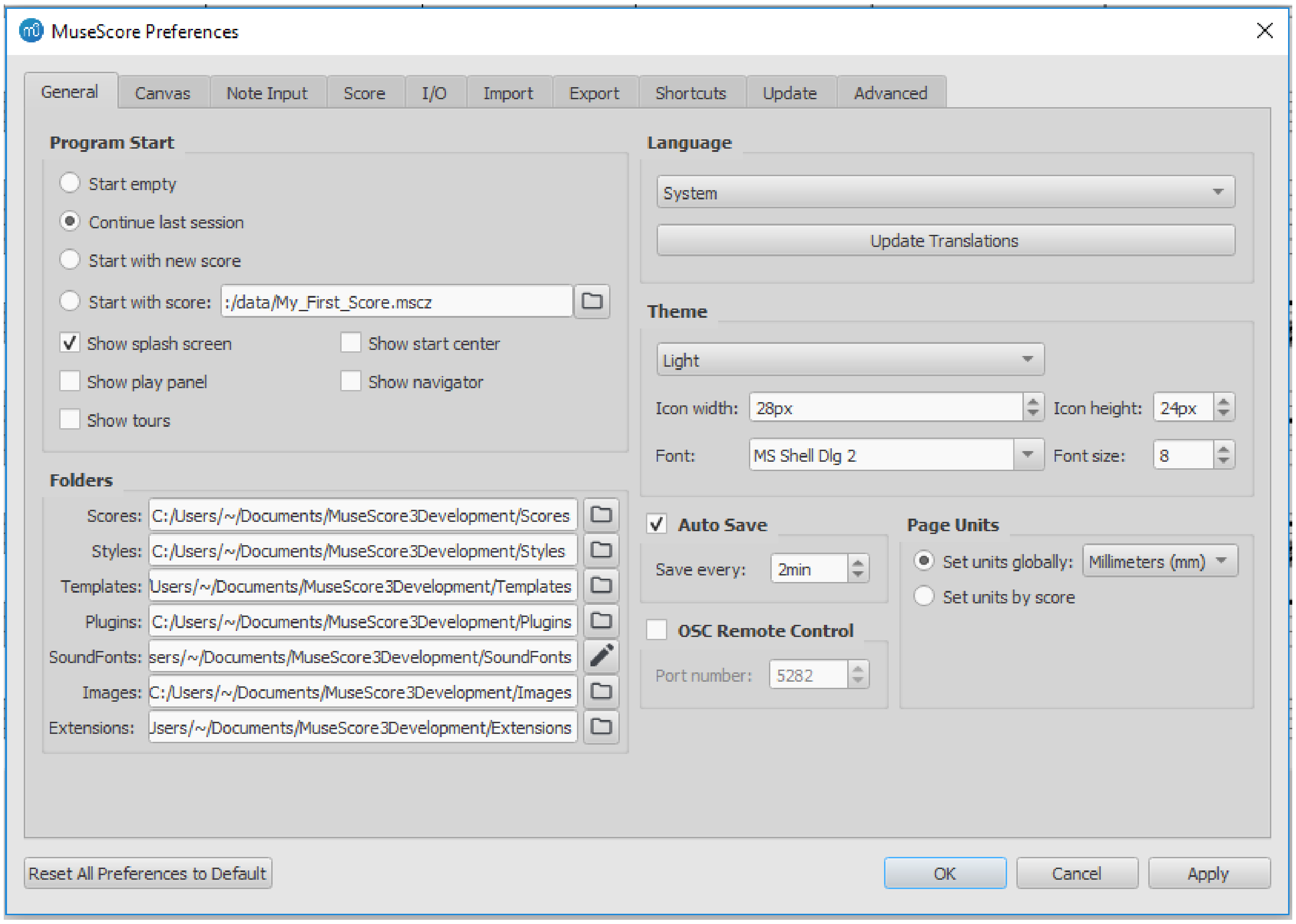
- A global user preference (MS Office, Apache Open Office)
- As a document property, stored in each file (Scribus, some Adobe apps)

MuseScore does neither of these. It defaults to millimeters at the start of every session, and only saves a user change of units for the duration of the session. This is completely non-standard, and not very user friendly.

I am proposing to change to not one, but a combination of both the standard options. It's not complex to implement or use. The most basic option requires adding a drop-down list of units to the Preferences dialog, on the General tab. I am proposing to create a pair of radio buttons:

- Set units globally
- Set units by score

Set units globally is accompanied by the aforementioned drop-down list. Here is a screenshot of the new design with a new Page Units group of options:



Program Start

- ( ) Start empty
- (●) Continue last session
- ( ) Start with new score
- ( ) Start with score: :/data/My_First_Score.mscz

- ✓ Show splash screen
- ☐ Show start center
- ☐ Show play panel
- ☐ Show navigator
- ☐ Show tours

Folders

Scores:	C:/Users/~/Documents/MuseScore3Development/Scores
Styles:	C:/Users/~/Documents/MuseScore3Development/Styles
Templates:	Users/~/Documents/MuseScore3Development/Templates
Plugins:	C:/Users/~/Documents/MuseScore3Development/Plugins
SoundFonts:	sers/~/Documents/MuseScore3Development/SoundFonts
Images:	C:/Users/~/Documents/MuseScore3Development/Images
Extensions:	Jsers/~/Documents/MuseScore3Development/Extensions

Language

System

Theme

Light

Icon width: 28px Icon height: 24px

Font: MS Shell Dlg 2 Font size: 8

✓ **Auto Save**

Save every: 2min

☐ **OSC Remote Control**

Port number: 5282

Page Units

- (●) Set units globally: Millimeters (mm)
- ( ) Set units by score

The list of units will be expanded to include all of the units in the QPageSize::Unit enumeration, adding: Points, Picas, Didot, and Cicero.

Dialog Box mscore/pagesettings.ui

If the user selects the Set units globally option, then the user will not have the ability to change the units in the page settings dialog. The dialog will use the appropriate text suffix when displaying numbers in those global units, but there will be no radio buttons or drop-downs for units.

On the other hand, if the user selects the Set units by score option, the user will select the units in the page settings dialog instead of the preferences dialog. The current pair of inches vs millimeters radio buttons will be replace by a drop-down list identical to the one in the preferences dialog, described above.

The Spatium, Rounding, XML storage, and Points @72dpi

There is an open issue about the rounding of the spatium. In its most basic form:

- Create a new score of any kind, and save it as an uncompressed .mscx file.
- That file contains a spatium value in millimeters, rounded to 5 decimals.
- Open the page settings dialog. The UI widget for spatium round to 3 decimals. Leave the spatium value unchanged at 1.764mm.
- Save the file again as uncompressed, and the XML spatium value is now 1.764.
- Open page settings and switch the units from millimeters to inches. Click OK.
- Save the file as uncompressed again. The XML value is now 1.753. Even though you have not changed the spatium value, it has changed in the file twice now due to two separate roundings.

You may scoff at the degree of precision in this error, but it is has serious, negative downstream consequences. The spatium serves three purposes:

1) It defines the distance between staff lines.

2) It is used as a unit of measure for many style settings.

3) It is used as a scaling factor for everything inside the body of the score, excluding titles, footers, and headers. That scaling factor is relative to the default value for the spatium, which is 5pt @72dpi, the standard definition of pt/points.

With #3, the size of these rounding errors is amplified because scaling is multiplying. There are also places in the code that compare the current spatium value to the default value to decide whether or not to do any scaling. SVG Export is one example. I believe PDF exporting is similar. *This scaling causes the actual font sizes for all sheet music* *elements to be out of sync with the point sizes specified in the Preferences dialog, which should be documented as part of the "edit the spatium" documentation*.

For any rounding issue, the obvious solution is to find a way to use ints instead of floats. That approach fits perfectly here, because the default value is already initialized as a whole number global constant `SPATIUM20`. The default spatium is 5pt in absolute terms, but the `SPATIUM20` variable multiplies that times `DPI_F` to arrive at `SPATIUM20 == 25`.

Thus the solution is to store the spatium in points @72dpi. Points are an immutable unit, like millimeters, and they are also fully integrated into the QPageSize and QPageLayout classes. MuseScore is already using both these classes in limited ways.

This connects to the UI changes because points are available as a unit in the new drop-down lists. As an emergency fallback, this allows the user to reset their spatium to the true default, no rounding. Today, once the rounding has occurred, the only way to fix it is to edit the uncompressed file.

Additionally, if the spatium is to be stored in points, then all the page settings numeric values can be stored in points too. They are currently stored in inches – the only values in the file stored in inches. The style values might be in points or in the currently specified units, global or by the current score. That's an open question that will be answered by modifying all the current code that converts these values anyway.

Units and Units Texts

Qt has infrastructure for page units. It even includes locale-specific, translated versions of standard page size names. But it does not include any kind of text for units.

MuseScore uses two strings for each unit: name and suffix/abbreviation. The name is only used in the two new drop-down lists. The suffix is used in all the numeric fields in the page settings dialog, and possibly in other places, but I'm not sure. Where else are non-staff space units displayed, besides font sizes in points?

Currently MuseScore only supports two units, millimeters and inches. The text for those units is hardcoded in pagesettings.cpp, maybe elsewhere too, all based on a boolean. The new implementation uses an integer index into the drop-down lists, which also aligns with the QPageSize::Unit enumeration. There is more text, and the structure of the code must become more systematized.

Based on the way other such things are setup in MuseScore, these could be implemented as new styles, but then the ids would not align with the Qt enumerations – unless it's acceptable to put them first in the MuseScore style id enum.

Source Code Changes

The UI changes are straightforward to implement. The further integration with QPageSize and QPageLayout offers more possibilities and pitfalls. Changing the page settings currently involves three layers:

1) The widgets in pagesettings.ui

2) The preview score's style object, specifically these ten styles:

```
Sid::pageWidth
Sid::pageHeight
Sid::pagePrintableWidth
Sid::pageEvenLeftMargin
Sid::pageOddLeftMargin
Sid::pageEvenTopMargin
Sid::pageEvenBottomMargin
Sid::pageOddTopMargin
Sid::pageOddBottomMargin
Sid::pageTwosided
```

3) The current score's style, which is updated from the preview score when you click Apply or OK.

Note that there are no right margin styles. This is because even/odd pages make them obsolete. evenRight == oddLeft, oddRight == evenLeft;

The page number offset is not cached as a style, but as the `_pageNumberOffset` member variable of class `Score`. QPageLayout does not support this property.

The other property not supported by QPageLayout is `Sid::pageTwosided`.

Because of even/odd pages, MuseScore will require two instances of QPageLayout, to cache all 8 margin values. Those two QPageLayouts will share an instance of QPageSize.

Initially it looks like a good idea to use QPageLayout/QPageSize to replace these style entries. But that has adverse consequences in several places where everything is currently handled by styles. It's very useful to cache the page settings in the styles, everywhere except inside pagesettings.cpp.

So I am implementing this as an additional layer used mostly inside the page settings dialog code. It will offload as many unit conversions as possible to Qt, which increase unit conversion reliability, and eliminate the need for some MuseScore conversion code/variables. Qt definitions of units will be used for both input and output, as often as possible. There will still be staff space units, which are user-specified and calculated at run time by MuseScore, as well as the 360dpi units for Qt painting (72dpi * 5).

I might end up implementing a `Score::pageLayout` property, but that would institutionalize the duplication of the style values. There is no perfectly elegant solution to this, they all add an additional layer of stuff somewhere. This score property would be useful in file.cpp, maybe elsewhere. Institutionalization might be the way to go. It would be another preview score to current score copy operation upon Apply. But you're copying only two pointers - the QPageSize is already contained by the QPageLayouts.

That implementation would mean that pagesettings.cpp would modify the page layouts in the preview score, then copy them to the current score. It is a question of putting these variables in class `Score` or in pagesettings.cpp.

I am going to opt for full institutionalization and put them in score.h. It makes them available in file.cpp and elsewhere. The margins and widths are used in 14 source files. I'll be cataloging all the changes...