Maintainer's guide to `xml2ly`

Jacques Menu

December 22, 2019 version

Abstract

This document presents the design principles and architecture os `xml2ly`, as well as the way to maintain it. It is part of the `libmusicxml2` documentation, to be found at https://github.com/grame-cncm/libmusicxml/tree/lilypond/doc.

In the `libmusicxml2` library, the source code specific to `xml2ly` can be found at https://github.com/grame-cncm/libmusicxml/tree/lilypond/src/lilypond and https://github.com/grame-cncm/libmusicxml/tree/lilypond/src/interface.

All the examples mentioned can be downloaded from https://github.com/grame-cncm/libmusicxml/tree/lilypond/files/samples/musicxml. They are grouped by subject in subdirectories, such as basic/HelloWorld.xml.

1 Acknowledgements

The scores fragments shown in this document have been produced by translating the '`.xml`' files to LilyPond syntax, and then creating the graphical score with LilyPond.

The translations have been done by `xml2ly`, a prototype tool developed by this author. `xml2ly` and some of the specific examples presented in this document are this author's contribution to `libmusicxml2`, an open-source C++ library created and maintained by Dominique Fober at Grame, Lyon, France. The home page to `libmusicxml2` is https://github.com/grame-cncm/libmusicxml.

The reader is invited to handle the '`.xml`' file examples with their own software tools to compare the results with the ones herein.

Tests with other score editing applications are mentioned in this document, namely Sibelius^{TM}, Finale^{TM} and MuseScore, which is open-source. `musicxml2ly` is mentioned too: this translator is supplied with LilyPond. This author doesn't own licenses for other commercial applications such as Dorico^{TM} or Capella^{TM}.

2 Overview of `xml2ly`

2.1 Why `xml2ly`?

MusicXML (*Music eXtended Markup Language*) is a specification language meant to represent music scores by texts, readable both by humans and computers. It has been designed by the W3C Music Notation Community Group (https://www.w3.org/community/music-notation/) to help sharing music score files between applications, through export and import mechanisms.

The homepage to MusicXML is https://www.musicxml.com.

MusicXML data contains very detailed information about the music score, and it is quite verbose by nature. This makes creating such data by hand quite difficult, and this is done by applications actually.

2.2 What `xml2ly` does

3 Prerequisites

In order to maintain `xml2ly`, one needs to do the following:

- obtain a working knowledge of C++ programming. The code base of `xml2ly` uses classes, simple and multiple inheritance, and templates;
- study MusicXML, starting maybe from `IntroductionToMusicXML.tex`. A deep knowledge of that matter comes with experience;
- study the architecture of `libmusicxml2`, which can be seen at `libmusicxmlArchitecture.pdf`, and is presented in figure 1. It shows the place of `xml2ly` in the whole.
-

4 Programming style and conventions

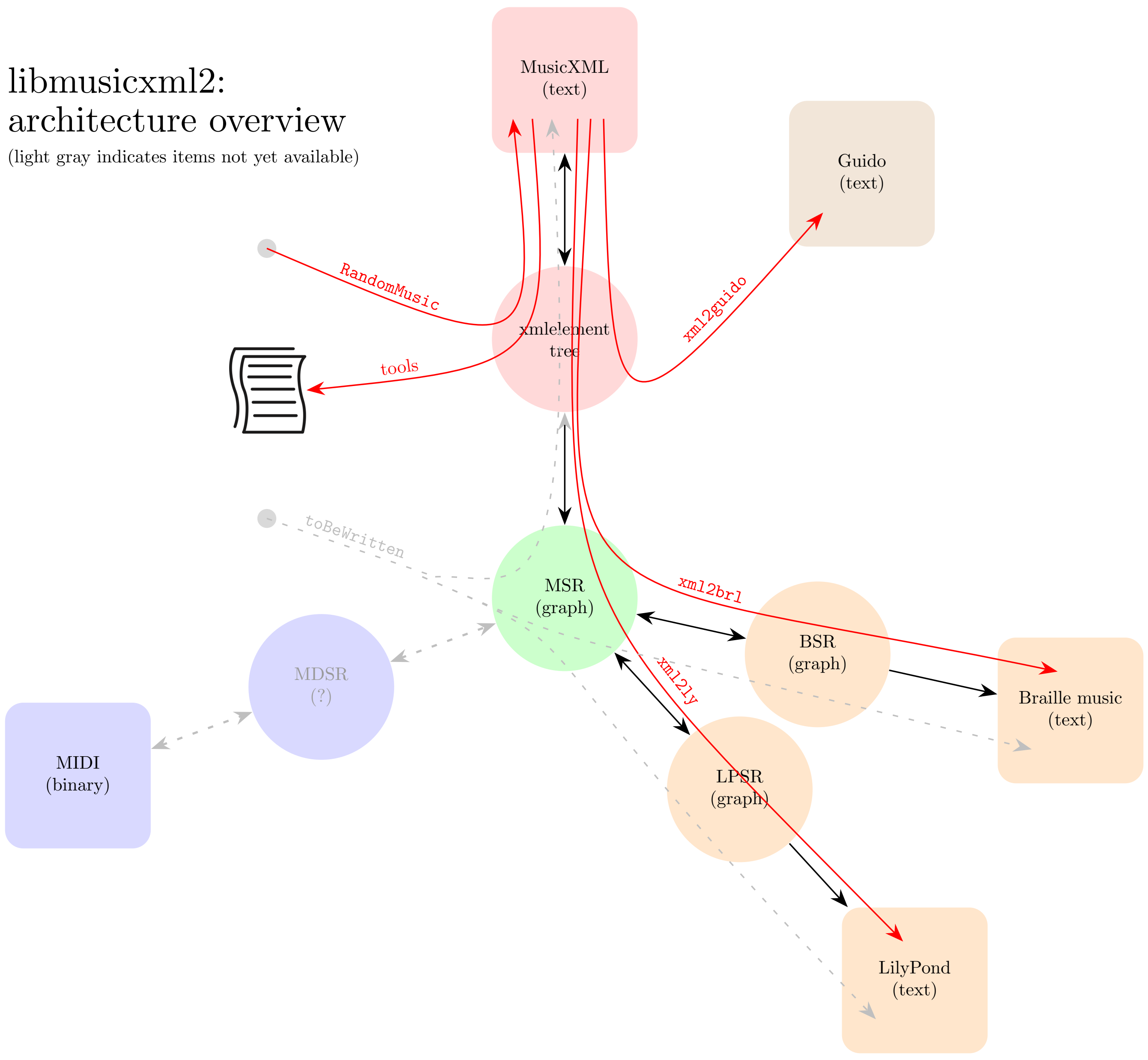
The following text-editing conventions are used:

- tabs are not used before the first non-space character in a line, two spaces are used instead;
- the code is not tightly packed: declarations in classes have the members' names aligned vertically, with many spaces before them if needed, and empty lines are used to separate successive activities in methods.

The code base of `xml2ly` is *defensive-programming* oriented, which means that:

- identifiers are explicit and long if needed – only very local ones are short, such as iteration loops indexes;
- the code is organized in sections, with an initial comment documenting what the code does.

Figure 1: `libmusicxml2` architecture



Entity	Description
xmlelement tree	a tree representing the MusicXML markups such as `<part-list>`, `<time>` and `<note>`
MSR	Music Score Representation, in terms of part groups, parts, staves, voices, notes, …
LPSR	LilyPond Score Representation, i.e. MSR plus LilyPond-specific items such as `\score` blocks
BSR	Braille Score Representation, with pages, lines and 6-dots cells
MDSR	MIDI Score Representation, to be designed
`RandomMusic`	generates an xmlelement tree containing random music and writes it as MusicXML
tools	a set of other demo programs such as `countnotes`, `xmltranspose` and `partsummary`
`toBeWritten`	should generate an MSR containing some music and write it as MusicXML, LilyPond and Braille music
`xml2ly`	performs the 4 hops from MusicXML to LilyPond to translate the former into the latter
`xml2brl`	performs the 4 hops from MusicXML to Braille music to translate the former into the latter (draft)

- Note: `xml2ly` has a '`-jianpu`' option
- Note: `midi2ly` translates MIDI files to LilyPond code
- Note: `lilypond` can generate MIDI files from its input

`xml2guido` v2.3, `xml2ly` v0.9, `xml2brl` v0.01, August 2019

Listings

Contents