Lilypond

A Guide To Open Source Music Notation

Eugene Cormier
Acadia University
February 14, 2015

Table of Contents

Preface – A comparison of open vs. closed source & notation "editors" vs. "formatters"

Lilypond is a powerful music typesetting program available for Windows, MacOS X, and Linux/Unix. It enables you to easily create and print music scores. Lilypond is also open source software which has many advantages over other proprietary (retail) software titles. Some of these advantages include:

Proprietary Notation

1. If there is an operating system update (like a new MS Windows/MacOS) and it won't run your old programs, it is at the whim of the company whether or not to develop a new version of the program. If they do you'll most likely have to upgrade at a cost.

2. Proprietary notation software companies, in their battle for market share and money, dumb down notation software in order to make it accessible to beginners.

3. The software thinks of the music as a picture with specific elements (notes/rests/clefs) at specific positions. This means:

   a) After entering the music there is usually a fair amount of editing necessary in order to make the score appear correct. (no overlapping items and correct spacing)

   b) over time as the program is updated, at a cost, your music will still look the same (or very similar) with individual elements at the same places.

Lilypond

1. If there is an operating system update, Lilypond is updated quickly due to the fact that it's not owned by any one individual, but a community. Anyone can pitch in and help create/improve the program.

2. Lilypond is directed by a community of users and programmers. If it doesn't do what you want it to, the way you want to do it, ask the developers and they'll put it in (try the same with your common off the shelf software). If more people ask for the same feature, it gets added much faster. And you can also make donations to speed the process as well. Lilypond does not care so much about $$ since it's free, so it's really guided by the will and needs of individuals in it's community.

3. Lilypond is not a program in which you point and click your notes. (Although there are many nice front-ends available which feel similar to proprietary notation programs) The music is stored in a text file as the elements of a score. Then you run Lilypond on the text file to produce a ps/pdf file. This is a superior entry method as the program makes its own decisions on element spacing when it is run. It considers all the elements on the page and figures the best positions for each element. This means it can't work like other notation programs, it would have to recalculate the entire score each time you enter a new note. But it also means:

   a) Since Lilypond looks at all the elements and then determines what it thinks is the best layout, there is normally little to no editing after input. Of course you can override most (or all) of Lilypond's automatic settings, but you shouldn't normally need to.

   b) Since Lilypond doesn't store locations of elements, as Lilypond is updated (which is of course always free), your scores will just look better, no extra work!

First off...

Lilypond uses text files as input.....let me get through some basics:

1. The easiest way to get going with Lilypond is to simply use: http://lilybin.com
The site allows you to get your feet wet with the language before having to install it. You can enter code, download MIDI/pdf files and see your error messages. The only place where it doesn't deliver is if you need to create scores and parts using multiple files.

2. There are many free front-ends (point & click interfaces) for Lilypond like: Denemo (http://denemo.sourceforge.net/), Noteedit (http://noteedit.berlios.de/), Gscore (http://www.gscore.org/). I find entering scores purely by code is actually much faster and with some practice allows for much more control. There are some programs/websites created to make plain text entry easier: Frescobaldi http://frescobaldi.org/

3. When editing the text files do not use a word processor (like MS Word or OpenOffice Writer) as these save their own formatting codes inside the documents. Lilypond won't understand these formatting codes and will be unable create the pdf. Instead you should use a text editor (in Linux I recommend using Kate or gedit) or if you use Windows you can download (for free) a very nice text editor like Notepad2. The reason for not just using the Notepad supplied with MS Windows is that from time to time, you will make mistakes, and when you run Lilypond, it will let you know exactly where you made your mistake (like 3:18 means: line 3, column 18) and programs like Kate and Notepad2 tell you the exact line and column number of the cursors current position. http://www.flos-freeware.ch/notepad2.html

4. When you save your Lilypond file, it's best to end the filename with .ly (in MS Windows this will enable a right click to give you the options of: 1) create pdf; 2) edit

5. If you use Linux I recommend running Lilypond from the command line. I just like to see everything that is happening (in MS Windows simply right clicking and "Create PDF" automatically brings up an error log if necessary)

6. When starting with Lilypond, I recommend making a small file with a mistake to see how precise Lilypond really is:

   `{ 4c }`   entering this code will result in the following error:

   ```
   GNU LilyPond 2.8.0
   Processing `test.ly'
   Parsing...
   test.ly:1:2: error: syntax error, unexpected DIGIT       ← this tells us the error is on line 1, column 2
   {                                                         ← this broken line show exactly
     4c }                                                    ← where the error starts
   test.ly:1:0: error: errors found, ignoring music expression
   { 4c }
   test.ly: 0: warning: no \version statement found,  add   ← Lilypond wants us to add
   \version "2.8.0"                                          ← this line for compatibility
   for future compatibility
   ```

7. You should always include a \version command at the top of your Lilypond files. Lilypond is updated quite frequently, and after a couple of years it might be hard to think back to which version of Lilypond you were using. Also Lilypond checks this to make sure the file is interpreted correctly.

8. For anything to print, it must be inside { }. Think of { } as an individual staff...the clef, time signature, notes, rests, barlines, etc all go on the staff, they all go inside the { }

9. When entering large works its best to include barline/bar number checks (Lilypond will tell you if theres something off making it easier to locate mistakes)

10. This guide uses the following conventions:

    Code This is used when I'm illustrating Lilypond code

    Tips This is used for special tips and tricks that extend basic functionality

11. You should always include comments in your scores so that you/others can understand what you did. There are 2 ways to make comments:
    a) single line comment. Just put a % on a line (anywhere), and Lilypond will ignore everything after that mark until the end of the line
    b) multi line comment. You start the comment with a %{ and Lilypond will ignore everything after that mark until you end the comment with a %}

12. There is complete online documentation at: http://lilypond.org

13. Remember all music must be inside { }

    Code
    c d e f g a b c   ← won't print anything
    { c d e f g a b c }   ← will print a c major scale

14. Relative pitches (in my opinion this is most often a better method; use \relative c{ }):
    the octave information is set at the beginning and each note's octave will be determined by proximity to the last note, unless you override that with the ' octave up, or , octave down commands (remember: the note will stay as close as possible to the previous note. If the first note is a "C" and the second note is a "F" it goes up; if the second note is a "G" it goes down – the "G" up is a P5, while the "G" down is only a P4)

    Code
    \relative c'' { c d e f g a b c }

    here is the octave information

15. Absolute pitches (just use { }):
    you must enter each note's octave information

    Code
    { c'' d'' e'' f' g' }

Entering Music

Clefs

The \clef command happens inside the { } and after it you can put: treble, bass, alto, tenor, percussion, tab, "treble_8", "bass^15" as well as others. If you don't enter a \clef command Lilypond will default to a treble clef. If you want to change clefs, just put the \clef command with the clef name at the desired location.

Code \clef treble

Tips If you do a change of clef Lilypond automatically makes the second clef smaller, to avoid this use the following command just before the \clef command:
\override Staff.Clef #'full-size-change = ##t
or to print no clef at all (put '\once' before it if you want only one clef to be transparent):
\override Staff.Clef #'transparent = ##t

Key Signatures

The \key command comes inside the { }. After the \key command comes the letter name of the key (ex: a=A, aes=A flat, ais=A sharp) and then comes the key type (ex: \major, \minor, \dorian, \mixolydian, etc...)

Code \key a \major   ← A major
\key aes \major ← A flat major

Tips To stop cautionary accidentals for key changes use the following command:
\set Staff.printKeyCancellation = ##f
or to print no key signature at all:
\override Staff.KeySignature #'transparent = ##t

Time Signatures

The \time command comes inside the { }. If you don't enter a \time command Lilypond will use the default which is common time "C".

Code \time 3/4

Tips If you'd like to use the numerical forms of 4/4 and 2/2 then use the following command just before the \time command:
\numericTimeSignature
to go back to normal:
\defaultTimeSignature
to have a pickup measure (the number is the duration of the pickup):
\partial 4   ← Partial 4 would be a quarter note pickup measure

Notes & Accidentals

Notes are written in absolute pitches (if the key signature is A major and you enter a g it is interpreted as G natural) regardless of key signature (this is actually a major benefit when   it   comes   to things like transposition).

Code \relative c { c d e f g a b c }   ← C major scale

Tips Remember: notes must be entered in { }
To write one or more notes in one voice put <> around the notes:
<c e>4
To add an arpeggio line to a chord:
<c e>4\arpeggio

To write one or more notes in different voices:
<<{ c4 } \\{ e4 }>>
To have differently dotted notes in different voices share the same note head:
\mergeDifferentlyDottedOn

To add accidentals use es for flat and is for sharp

Code

bes   ← B flat
fis   ← F sharp
\relative c { d e fis g a b cis d } ← D major scale
\relative c { f g a bes c d e f }   ← F major scale

Tips

You can use ! to force an accidental and ? for an accidental in brackets (complimentary)
bes!   ← B flat – the flat will appear regardless of the key signature
fis?   ← F sharp – the sharp will be printed inside brackets

Tips

To hide/unhide notes use:
\hideNotes or \unHideNotes
To make a note(s) grace note(s) use:
\grace   { c d e }

Tips

You can force lilypond to use english accidentals (instead of dutch):
\language "english"
Now you can add an 's' to a note to make it sharp and a 'f' to a note to add a flat
\relative c { d e fs g a b cs d }   ← D major scale
\relative c { f g a bf c d e f }   ← F major scale

Rests & Invisible Spacers

Rests are entered like notes except you use r

Code

r2   ← a half rest

Tips

To get a full measure rest:
R1   R2.   ← or whatever value to fill the measure
This will make it look like a whole rest, but only use as many beats as you assign it
R1 will fill a 4/4 measure, R2. Will fill a 3/4 measure

To do multi-measure collapsible rests:
R1*4   ← 4 measures of whole rests
And the command to collapse them:
\compressFullBarRests
To get longer rests:
r\maxima   ← equal to 4 breves
r\longa   ← equal to 2 breves
r\breve
To control vertical spacing of rests write the note you want followed by the rest
command:
d4\rest

Invisible spacers are entered like notes except you use s

Code

s2   ← an invisible half note

Duration

The length of the note/rest/spacer is set using numbers:

1=whole 2=half 4=quarter 8=eighth etc... (or \maxima, \longa, \breve

g1   ← G whole note
g\breve ← G breve
g16   ← G 16th note

Once you've entered a duration subsequent notes/rests/spacers entered without a duration will assume the same duration
b8 b b b   ← will print 4 eighth note "b"s

Caution: remember to start each line/section with a duration attached to the note whether or not it is needed. This will save you trouble if you want to copy/paste the section later.

Dotted Durations

Use one or two periods (when you use a dot <u>you must include the duration number</u>)

b2.   ← B dotted half note
a4..   ← A double-dotted quarter note

To force the placement for the dots use:
\dotsUp   ← dots go above line
\dotsDown   ← dots go below line
\dotsNeutral   ← lilypond determines placement

Tuplets

To create triplets use \times 2/3. Remember: you only need to add this once before a string of triplets; and the bottom number is the one which will be printed. So \times 4/5 would print a 5 over the tuplet.

\times 2/3 { c8 c c c c c }   ← will print 2 sets of eighth note triplets

\times 2/3   ← creates a triplet, '3' notes in the space of '2'
\times 4/5   ← creates a quintuplet, '5' notes in the space of '4'
\relative c' { \tuplet 3/2 4 { a8 b c d e f g a b } } ← multiple triplets in a row

Stem Directions

Stem directions can be changed for a single note or for any number of notes using \stemDown, \stemUp and \stemNeutral. (remember this is done automatically when using multiple voices)

c'''\stemUp   ← high c with the stem up
\stemUp c d e   ← c d and e all stem up

Beaming

[ ]   ← these start and stop beams
a8 a[a a]   ← the first note will not be beamed with the others

** remember to always put the beam start [or stop] after the note you want it on
c[d e f]   ← will beam from the C to the F, <u>not</u> from the D to the F

Barlines

Barlines are automatically added by Lilypond. Their behavior can be tuned with:

Code

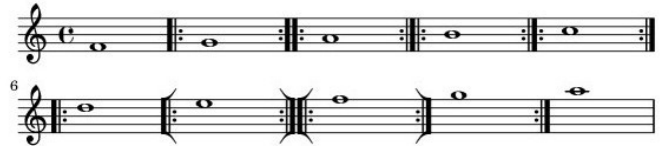\bar "" ← remove one automatic barline
\bar "|" ← add a barline

Tips

To remove all barlines use:
\override BarLine #'transparent = ##t
There are many kinds of barlines

```
f1 \bar "|"
f1 \bar "."
g1 \bar "||"
a1 \bar ".|"
b1 \bar ".."
c1 \bar "|.|"
d1 \bar "|."
e1
```

```
f1 \bar ".|:"
g1 \bar ":..:"
a1 \bar ":|.|:"
b1 \bar ":|.:"
c1 \bar ":.|.:"
d1 \bar "[|:"
e1 \bar ":|][|:"
f1 \bar ":|]"
g1 \bar ":|."
a1
```


Remember: you only need to add the type of barline once in a score and it will cross all staves.

Repeats

Normal repeats

Code

\repeat volta 2 { c4 d e f }   ← the '2' is the number of repeats

Normal repeats with alternate endings

Code

```
\repeat volta 2 { c4 d e f | }
        \alternative {
                { c2 e | }        1st ending
                { f2 g | }        2nd ending
                }
```

Text

Headers

The following are some of the possible headers (automatically formatted text):

Code
```
\header {
        title = "title"
        subtitle = "subtitle"
        composer = "composer"
        arranger = "arranger"
        instrument = "instrument"
        opus = "opus"
        piece = "piece"
        poet = "poet"
}
```

And heres how they look:

<p align="center">title</p>
<p align="center">subtitle</p>

poet	**instrument**	composer
		arranger
piece		opus

Remember: since it sets up page preferences, the \header command comes before the \relative c command and not inside { }

Tips
You can also use the following in the header section:
copyright   ← prints at the bottom of page 1
tagline ← prints at the bottom of the last page

Other Text

Code
\markup { hello world }   ← print text between staves (this comes outside { })
c^\markup { hello world }   ← print text above a note (^ means above)
r_"hello world"   ← print text below a rest. Notice the \markup command is not
                                   needed unless you need to use a command (_ means below)

Tips
You can use flat/sharp signs in text by using the \flat and \sharp commands
\markup { write a C \sharp major scale }

There are certain characters that Lilypond uses for code that you might want to have printed as text (like a "#"), in this case all you have to do is put the text in quotation marks
\markup { "#1. Write a C major scale" }
\markup { "#"1. Write a C \sharp major scale }

You can't have a command inside " " - the \sharp command inside " " will make Lilypond fail. You can connect text to any note/rest/spacer

Lyrics

To use lyrics simply add a \addlyrics { } after the music { }

\addlyrics { hello world __ } ← If a single syllable word is sung over multiple
notes add 2 underscores to create a line

\addlyrics { hel -- lo world } ← If a multiple syllable word is sung over multiple
notes add 2 dashes to create a hyphen

\addlyrics { hello world } ← To add multiple lines of lyrics
\addlyrics { line two }
\addlyrics { line three }

MIDI

To get Lilypond to output MIDI, put all music inside a \score { } and then after that put a \midi { } command
if you want the pdf and the MIDI files to be made don't forget to add a \layout { }

```
\score {
        …music…
        \layout { }
        \midi { }
}
```

Variables

Variables can be a very powerful tool. They can change multiple elements using a single command. I'll offer two, of many, possibilities:

If you want to enter a note pattern that is recurring, you can use variables to ease the process. Here's a small example of how to use it: first you create a variable and give it some notes and then use it as a command in your {}

test = { e d e c } ← the variable "test" and it's notes "e d e c"
{ \test \test \test } ← calling the variable 3 times

Another way to use variables would be if you wanted to create a score and each instrument was going to have the same key signature, time signature and tempo indication

test = { \key a \major \time 4/4 \tempo adagio }
then you could add the command \test at the beginning of each part. Also if you
needed to change the key to G major, you would change it in the variable, and all
instances would be immediately updated
Here are some variables that I commonly use:
notimesig = { \override Staff.TimeSignature #'transparent = ##t }
nokeysig = { \override Staff.KeySignature #'transparent = ##t }
nokeycancel = { \set Staff.printKeyCancellation = ##f }
nobarlines = { \override Staff.BarLine #'transparent = ##t }
nobarlinenumbers = { \override Score.BarNumber #'transparent = ##t }
noclefresize = { \override Staff.Clef #'full-size-change = ##t }
noclef = { \override Staff.Clef #'transparent = ##t }
nonoteheads = { \override NoteHead #'transparent = ##t }
nostems = { \override Stem #'transparent = ##t }
nobeams = { \override Beam #'transparent = ##t }

Joining Staves

You just add the \new Staff command inside of << >>

```
<<
        \new Staff { s1 }
        \new Staff { s1 }
>>
```

Grand Staff & Staff groups

```
\new PianoStaff  <<
        \new Staff { s1 }
        \new Staff { \clef bass s1 }
>>
```
← Staves are connected with a curly brace

Instead of PianoStaff you can also use:
```
\new StaffGroup << >>
```
← Orchestral score
```
\new ChoirStaff  << >>
```
← Choir score (barlines do not extend across staves)

Creating a Score (and transposition)

Remember to keep all files in the same folder

1. Create a text file to contain the concert pitch notes called horn-notes.ly
    Add the following code to the file and save it:

    ```
    horn = \relative c { c d e f g }
    ```
    ← this sets the horn notes to the variable "horn"

2. To print the horn part:
    Create a text file to tell Lilypond how to print the notes so that they're readable to a horn player. Call it horn-part.ly
    Add the following code to the file and save it:

    ```
    \include "horn-notes.ly"
    \header { instrument = "French Horn" }
    { \transpose f c' { \horn } }
    ```
    ← this loads in the horn-notes file
    ← this prints "French Horn" on the top
    ← this transposes and prints the part

    Remember: when transposing from concert pitch for an instrumentalist, the instruments pitch goes first then "c" (the French Horn is an "F" instrument, therefore the code is: \transpose f c { }

3. To print the score:
    Create a text file to tell Lilypond how to print the score. Call it score.ly
    Add the following code to the file and save it:

    ```
    \include "horn-notes.ly"
    \include "clarinet-notes.ly"
    \include "trumpet-notes.ly"
    \new StaffGroup <<
            \new Staff \horn
            \new Staff \clarinet
            \new Staff \trumpet
    >>
    ```
    ← this loads in the horn-notes file
    ← this loads a file clarinet-notes
    ← this loads a file trumpet-notes
    ← this brackets the staves together
    ← a new staff with the variable horn
    ← a new staff with the variable clarinet
    ← a new staff with the variable trumpet
    ← this ends the brackets

Multi-Measure Rests in a Score

To create multi-measure (collapsible) rests replace the normal rests with:

Code R1*4   ← R = collapsible rests, 1 = how many beats per bar-so 1 means
whole or 4 beats, *4 = how many measures

Tips In the previous score example in the file "horn-part.ly" you would have:
\include "horn-notes.ly"   ← this loads in the horn-notes file
\header { instrument = "French Horn" } ← this prints "French Horn" on the top
{ \transpose f c' { \compressFullBarRests \horn } }
and that will make the rests collapse in the horn part, but still fully print in the score

Printing/Formatting Commands

The following codes can be added to the \paper { } command. The \paper { } command should come at the beginning of your Lilypond file. (this shouldn't come inside any { })

Paper size

a6, a5, a4, a3, legal, letter, 11x17

#(set-paper-size "letter")

Printing page numbers

print-page-number = ##f

Portrait/Landscape

#(set-paper-size "letter" 'landscape)

Resizing Music

#(set-global-staff-size 18)

Justify Music to bottom of page

t = "true" not justified; f = "false" justified

ragged-last-bottom = ##t
ragged-last-bottom = ##f

Justify Music to right side of page

t = "true" not justified; f = "false" justified

ragged-right=##t
ragged-right=##f

Indent first staff (how to turn off)

indent = 0

Margins

line-width = 7.5\in
left-margin = 0.45\in
bottom-margin = 0.25\in
top-margin = 0.25\in

Appendix A: Syntax

Syntax in Lilypond is very important: while { aes' } will print an A flat note up an octave, { a'es } will result in Lilypond not even being able to create the pdf

The overall syntax of a Lilypond event is as follows:

```
aes'8.[(\f^\markup { hi there }
12 345678 9
```

1. Note Name: this can be a letter for a note, a r for a rest or a s for a spacer

2. Accidentals: es = flat, is = sharp, nothing means natural

3. Octave Shift: ' = up an octave, , = down an octave

4. Duration: 1 = whole, 2 = half, 4 = quarter, 8 = eighth, 16 = sixteenth, etc...

5. Dotted: after the number (duration) you can enter a period (or 2) to make the duration dotted (or double-dotted)

*** After this point the order doesn't matter for the rest. For example: beaming could come after the slur or dynamics.

6. Beaming: the square brackets force beams on or off. [= start beam,] stop beam

7. Slurs: the (starts a slur,) stops the slur

8. Dynamics: you can use just about anything \fff

9. Text: this is how you attach text to a note. The ^ means above the note, _ means below the note. Just put your own text between the { }

Appendix B: Example - Putting it all together

```
%{
The following code is an example of where I put things and in which order. To help illustrate syntax, I've tried to include as much as
possible. Normally, Lilypond files do not need all these commands. (see Appendix C) Remember that the version # should match the
version of Lilypond that you use.
%}

\version "2.18.2" % this should be the version of Lilypond that you use

\header {
        copyright = "Eugene Cormier"
        title = "Lilypond Test"
        }

\paper {
         #(set-paper-size "legal" 'landscape)       % set paper size to legal landscape
        ragged-last-bottom = ##t                    % turns of verticle justify
        line-width = 12.5\in                        % these four lines are for margins
        left-margin = 0.45\in
        bottom-margin = 0.25\in
        top-margin = 0.25\in
        }
#(set-global-staff-size 18)                 % set staff size to 18

% assorted text
\markup { This is an example of Lilypond's commands }
\markup { To print a number sign "#" remember to put it in quotations marks }
\markup { To print sharp or flat signs with text use the \sharp \flat commands }

test = \new PianoStaff << % this creates a piano staff with a brace and sets it as a variable "test"
        \new Staff {       % the top staff
                \relative c" {
                        \clef treble \key aes \major          % clef and key
                        \numericTimeSignature                 % this prints numerical time signatures
                        \time 4/4
                        cis!^\markup { text above note } des?_\markup { text below note } e2 \bar "|:"
                        \once \override Staff.Clef #'full-size-change = ##t
                        \clef bass
                        f,,,4. g8 a b c\stemUp
                        r4 r s_\markup { text below empty space } r^\markup { text above rest }
                        }
                }
        \addlyrics { hello world }
        \new Staff {                                  % bottom staff
                \relative c" { \clef bass c d e }     % don't forget the bass clef
                }
        >>

{ \test }
```

Appendix C: Simple Example

```
%{
This example takes advantage of defaults. Default clef is treble, default time sig is 4/4, default key is C major and auto
barlines
%}

\relative c{
        e4 e f g
        g f e d
        c c d e
        e4. d8 d2^\markup { Beethoven }
        }
```

Appendix D: Lilypond Cheat Sheet

(This is taken from Lilypond's online documentation for your convenience)

Syntax	Description	Example
1 2 8 16	durations	
c4. c4..	augmentation dots	
c d e f g a b	scale	
fis bes	alteration	
\clef treble \clef bass	clefs	
\time 3/4 \time 4/4	time signature	
r4 r8	rest	
d~ d	tie	
\key ees \major	key signature	
a a'	raise octave	
c c,	lower octave	

c(d e)	slur	
c\(c(d) e\)	phrasing slur	
a8[b]	beam	
<< \new Staff ... >>	more staves	
c-> c-.	articulations	
c\mf c\sfz	dynamics	
a\< a\! a	crescendo	
a\> a a\!	decrescendo	
<c e>	chord	
\partial 8	pickup	

\times 2/3 {f g a}   triplets

\grace   grace notes

\lyricmode { twinkle }   entering lyrics
\new Lyrics   printing lyrics twinkle

twin -- kle   lyric hyphen

\chordmode { c:dim f:maj7 }   chords

\context ChordNames   printing chord names

<<{e f} \\{c d}>>   polyphony

s4 s8 s16   spacer rests