

LilyPond

El gravador de música

Manual d'aprenentatge

L'equip de desenvolupadors del LilyPond

Aquest fitxer ofereix una introducció al programa LilyPond versió 2.19.16.

Per a més informació sobre la forma en la qual aquest manual es relaciona amb la resta de la documentació, o per llegir aquest manual en altres formats, consulteu [Secció “Manuals” in Informació general](#).

Si us falta algun manual, trobareu tota la documentació a <http://www.lilypond.org/>.

Copyright © 1999–2015 pels autors.

La traducció de la següent nota de copyright s'ofereix com a cortesia per a les persones de parla no anglesa, però únicament la nota en anglès té validesa legal.

The translation of the following copyright notice is provided for courtesy to non-English speakers, but only the notice in English legally counts.

S'atorga permís per copiar, distribuir i/o modificar aquest document sota els termes de la Llicència de Documentació Lliure de GNU, versió 1.1 o qualsevol posterior publicada per la Free Software Foundation; sense cap de les seccions invariants. S'inclou una còpia d'aquesta llicència dins de la secció titulada “Llicència de Documentació Lliure de GNU”.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.1 or any later version published by the Free Software Foundation; with no Invariant Sections. A copy of the license is included in the section entitled “GNU Free Documentation License”.

Per a la versió del LilyPond 2.19.16

Índex General

1	Tutorial	1
1.1	Compilació d'un fitxer	1
1.1.1	Escriptura del codi d'entrada	1
	Generació del resultat	1
1.1.2	MacOS X	2
1.1.3	Windows	6
1.1.4	Línia d'ordres	11
1.2	Com escriure fitxers d'entrada	12
1.2.1	Notació senzilla	12
	Altures	12
	Duracions (valors rítmics)	14
	Silencis	15
	Indicació de compàs	15
	Indicacions de tempo	15
	Clau	16
	Tot a l'hora	16
1.2.2	Treball sobre els fitxers d'entrada	16
1.3	Gestió dels errors	18
1.3.1	Consells generals de solució de problemes	18
1.3.2	Alguns errors comuns	18
1.4	Com llegir els manuals	18
1.4.1	Material omès	18
1.4.2	Exemples amb enllaç	18
1.4.3	Panoràmica dels manuals	19
2	Notació corrent	20
2.1	Notació en un sol pentagrama	20
2.1.1	Línies divisòries i comprovacions de compàs	20
	Línies divisòries	20
	Comprovacions de compàs	20
2.1.2	Alteracions accidentals i armadures	20
	Alteracions accidentals	21
	Armadures	21
	Advertiment: armadures i altures	21
2.1.3	Lligadures d'unió i d'expressió	22
	Lligadures d'unió	22
	Lligadures d'expressió	22
	Lligadures de fraseig	23
	Advertiments: lligadures d'expressió en front a lligadures d'unió	23
2.1.4	Articulacions i matisos dinàmics	23
	Articulacions	23
	Indicacions de digitació	23
	Matisos dinàmics	24
2.1.5	Addició de text	24
2.1.6	Barres automàtiques i manuals	25
2.1.7	Instruccions rítmiques avançades	25
	Compàs parcial	26
	Grups especials	26

Notes d'adorn	26
2.2 Diverses notes a la vegada	26
2.2.1 Explicació de les expressions musicals	27
Analogia: expressions matemàtiques	27
Expressions musicals simultànies: diversos pentagrames	27
Expressions musicals simultànies: un sol pentagrama	28
2.2.2 Diversos pentagrames	28
2.2.3 Grups de pentagrames	29
2.2.4 Combinar notes per formar acords	30
2.2.5 Polifonia a un sol pentagrama	31
2.3 Cançons	31
2.3.1 Elaborar cançons senzilles	31
2.3.2 Alineació de la lletra a una melodia	32
2.3.3 Lletra en diversos pentagrames	35
2.4 Retocs finals	36
2.4.1 Organitzar les peces mitjançant variables	36
2.4.2 Afegir títols	38
2.4.3 Noms de nota absoluts	38
2.4.4 Més enllà del tutorial	40
3 Conceptes fonamentals	41
3.1 Com funcionen els fitxers d'entrada del LilyPond	41
3.1.1 Introducció a l'estructura dels fitxers del LilyPond	41
3.1.2 La partitura és una (única) expressió musical composta	43
3.1.3 Niuat d'expressions musicals	46
3.1.4 Quant a la impossibilitat de niuar claudàtors i lligadures	47
3.2 Les veus contenen música	48
3.2.1 Escolto veus	48
3.2.2 Veus explícites	53
3.2.3 Veus i música vocal	56
3.3 Contextos i gravadors	59
3.3.1 Explicació dels contextos	59
3.3.2 Creació de contextos	60
3.3.3 Explicació dels gravadors	62
3.3.4 Modificar les propietats dels contextos	63
3.3.5 Afegir i eliminar gravadors	68
3.4 Extensió de les plantilles	70
3.4.1 Soprano i violoncel	71
3.4.2 Partitura vocal a quatre veus SATB	74
3.4.3 Crear una partitura partint de zero	79
3.4.4 Estalviar tecleig mitjançant variables i funcions	84
3.4.5 Partitures i particel.les	86
4 Tweaking output	88
4.1 Tweaking basics	88
4.1.1 Introduction to tweaks	88
4.1.2 Objects and interfaces	88
4.1.3 Naming conventions of objects and properties	89
4.1.4 Tweaking methods	89
The <code>\override</code> command	89
The <code>\revert</code> command	90
The <code>\once</code> prefix	90
The <code>\overrideProperty</code> command	91

The <code>\tweak</code> command	91
The <code>\single</code> prefix	93
4.2 The Internals Reference manual	94
4.2.1 Properties of layout objects	94
4.2.2 Properties found in interfaces	98
4.2.3 Types of properties	99
4.3 Appearance of objects	100
4.3.1 Visibility and color of objects	100
The <code>stencil</code> property	100
The <code>break-visibility</code> property	102
The <code>transparent</code> property	102
The <code>color</code> property	103
4.3.2 Size of objects	105
4.3.3 Length and thickness of objects	108
4.4 Placement of objects	109
4.4.1 Automatic behavior	109
4.4.2 Within-staff objects	110
The <code>direction</code> property	111
Fingering	112
4.4.3 Outside-staff objects	114
The <code>outside-staff-priority</code> property	114
The <code>\textLengthOn</code> command	117
Dynamics placement	118
Grob sizing	118
4.5 Vertical spacing	119
4.6 Collisions of objects	123
4.6.1 Moving objects	123
4.6.2 Fixing overlapping notation	126
The <code>padding</code> property	126
The <code>right-padding</code> property	127
The <code>staff-padding</code> property	127
The <code>self-alignment-X</code> property	128
The <code>staff-position</code> property	128
The <code>extra-offset</code> property	128
The <code>positions</code> property	129
The <code>force-hshift</code> property	130
4.6.3 Real music example	131
4.7 Further tweaking	139
4.7.1 Other uses for tweaks	139
Tying notes across voices	139
Simulating a fermata in MIDI	139
4.7.2 Using variables for layout adjustments	141
4.7.3 Style sheets	142
4.7.4 Other sources of information	146
4.7.5 Advanced tweaks with Scheme	147

Annex A	Templates	149
A.1	Built-in templates	149
A.2	Single staff templates	152
A.2.1	Notes only	152
A.2.2	Notes and lyrics	152
A.2.3	Notes and chords	153
A.2.4	Notes, lyrics, and chords	153
A.3	Piano templates	154
A.3.1	Solo piano	154
A.3.2	Piano and melody with lyrics	155
A.3.3	Piano centered lyrics	156
A.4	String quartet templates	157
A.4.1	String quartet	157
A.4.2	String quartet parts	158
A.5	Vocal ensembles templates	161
A.5.1	SATB vocal score	161
A.5.2	SATB vocal score and automatic piano reduction	163
A.5.3	SATB with aligned contexts	165
A.5.4	SATB on four staves	166
A.5.5	Solo verse and two-part refrain	168
A.5.6	Hymn tunes	170
A.5.7	Psalms	172
A.6	Orchestral templates	175
A.6.1	Orchestra, choir and piano	175
A.7	Ancient notation templates	178
A.7.1	Transcription of mensural music	178
A.7.2	Gregorian transcription template	184
A.8	Other templates	184
A.8.1	Jazz combo	184
Annex B	GNU Free Documentation License	191
Annex C	Índex del LilyPond	198

1 Tutorial

Aquest capítol ofereix una introducció bàsica al treball amb el Lilypond.

1.1 Compilació d'un fitxer

Aquesta secció presenta el concepte de “compilació”: el processament dels documents d'entrada del LilyPond (escrits per vos mateix) per produir fitxers de sortida.

1.1.1 Escriptura del codi d'entrada

“Compilació” és una paraula que significa processar un text d'entrada en format del LilyPond per produir un fitxer que es pot imprimir i (de manera opcional) un fitxer MIDI que es pot reproduir. El primer exemple mostra l'aspecte d'un senzill fitxer de text d'entrada.

Aquest exemple mostra un fitxer d'entrada senzill:

```
\version "2.19.16"
{
  c' e' g' e'
}
```

El resultat té aquest aspecte:



Nota: la música i la lletra escrita al codi d'entrada del Lilypond ha d'anar sempre entre **{ claudàtors }**. Els claudàtors haurien també d'estar rodejats per espais a no ser que es trobin al principi o al final d'una línia, per evitar ambigüitats. És possible que s'ometen en alguns exemples del manual actual, però eviteu d'ometre-les a la vostra pròpia música! Per veure més informació sobre la presentació dels exemples del manual, consulteu [Secció 1.4 \[Com llegir els manuals\]](#), [pàgina 18](#).

A més, l'entrada del Lilypond és **sensible a les majúscules**. ‘{ c d e }’ és una entrada vàlida; ‘{ C D E }’ produeix un missatge d'error.

Generació del resultat

El mètode per produir un resultat imprès depèn del nostre sistema operatiu i del programa o programes que utilitzem.

- [Secció 1.1.2 \[MacOS X\]](#), [pàgina 2](#) [Secció 1.1.2 \[MacOS X\]](#), [pàgina 2](#) (gràfic)
- [Secció 1.1.3 \[Windows\]](#), [pàgina 6](#) [Secció 1.1.3 \[Windows\]](#), [pàgina 6](#) (gràfic)
- [Secció 1.1.4 \[Línia d'ordres\]](#), [pàgina 11](#) [Secció 1.1.4 \[Línia d'ordres\]](#), [pàgina 11](#) (consola)

Hi ha diversos altres editors de text amb un suport específic a l'edició de text del LilyPond. Per veure més informació, consulteu [Secció “Entorns millorats” in Informació general](#).

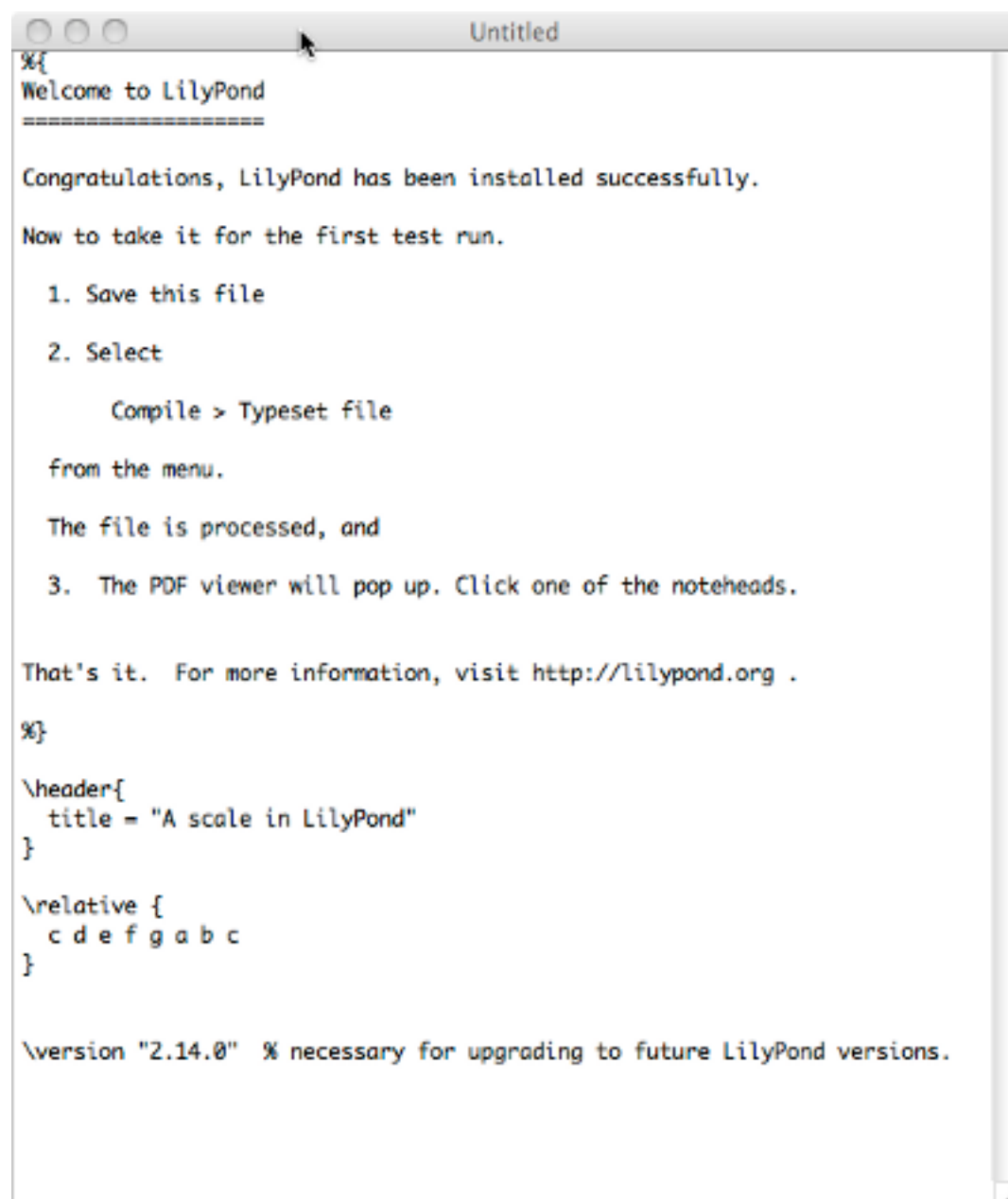
Nota: El primer cop que executeu el LilyPond, trigarà un minut o dos perquè totes les tipografies del sistema han de ser analitzades prèviament. Després d'això, el LilyPond serà molt més ràpid!

1.1.2 MacOS X

Nota: These instructions assume that you are using the LilyPond application. If you are using any of the programs described in *Secció “Easier editing”* in *Informació general*, consult the documentation for those programs should you have any problems.

Step 1. Create your ‘.ly’ file

Double click the LilyPond.app, an example file will open.



```
%{
Welcome to LilyPond
=====

Congratulations, LilyPond has been installed successfully.

Now to take it for the first test run.

  1. Save this file

  2. Select

      Compile > Typeset file

  from the menu.

  The file is processed, and

  3. The PDF viewer will pop up. Click one of the noteheads.

That's it.  For more information, visit http://lilypond.org .

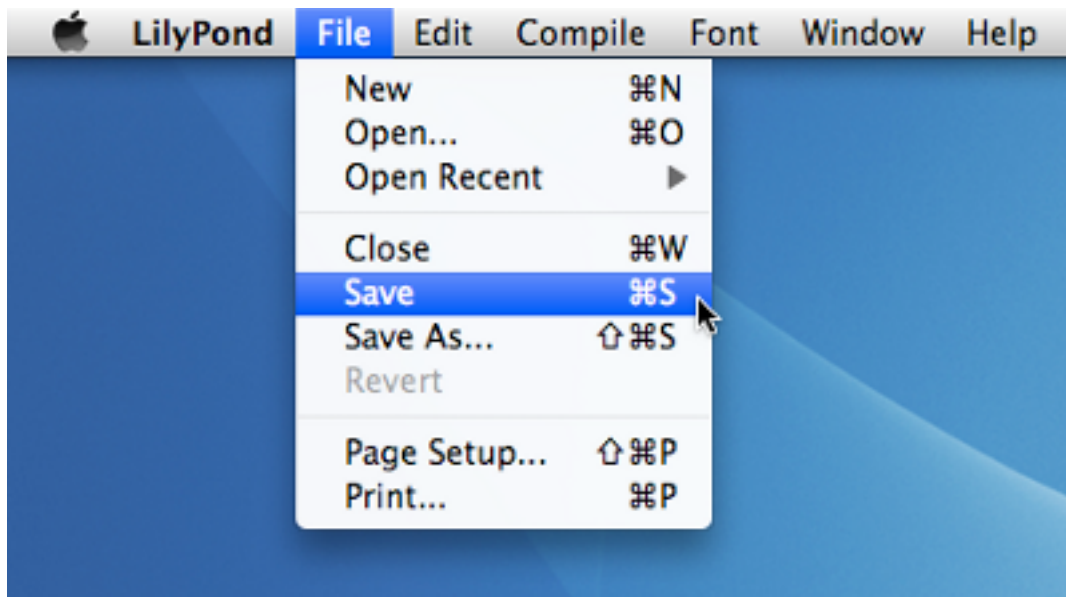
%}

\header{
  title = "A scale in LilyPond"
}

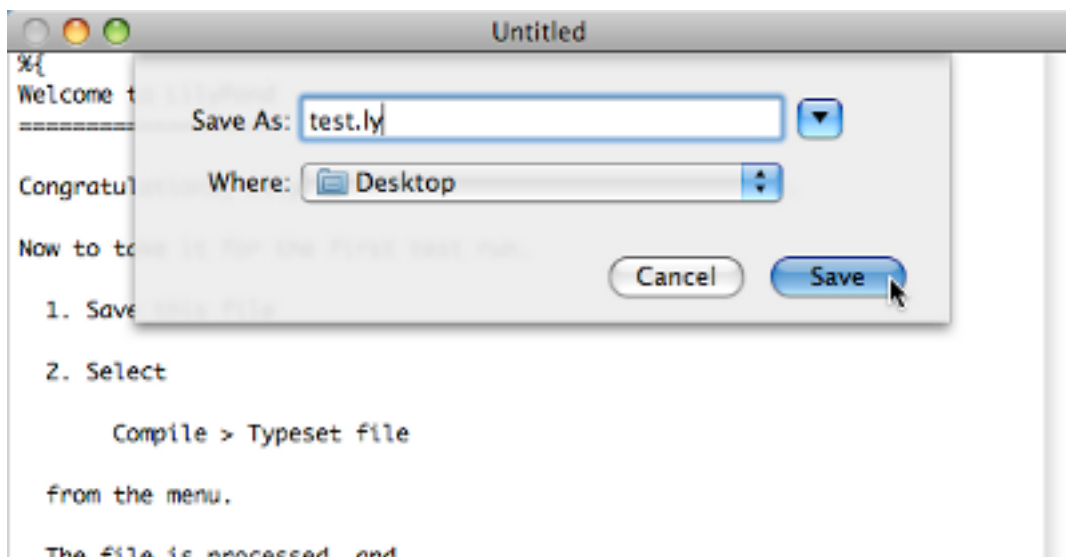
\relative {
  c d e f g a b c
}

\version "2.14.0" % necessary for upgrading to future LilyPond versions.
```

From the menus along the top left of your screen, select **File > Save**.

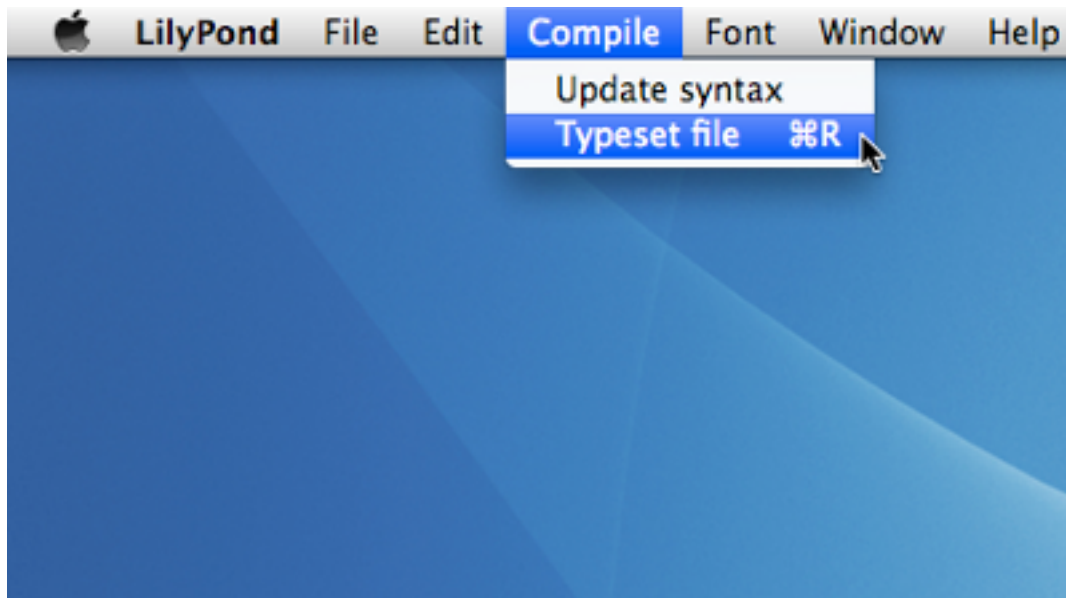


Choose a name for your file, for example 'test.ly'.

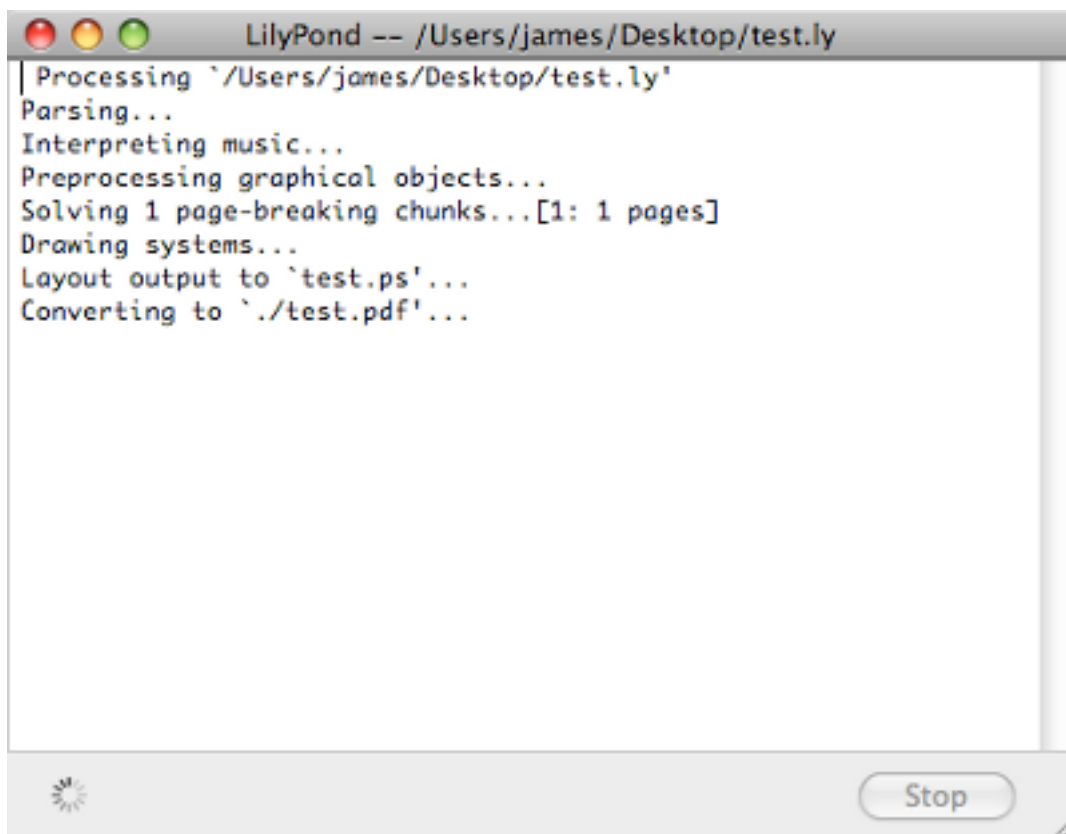


Step 2. Compile (with LilyPad)

From the same menus, select Compile > Typeset.

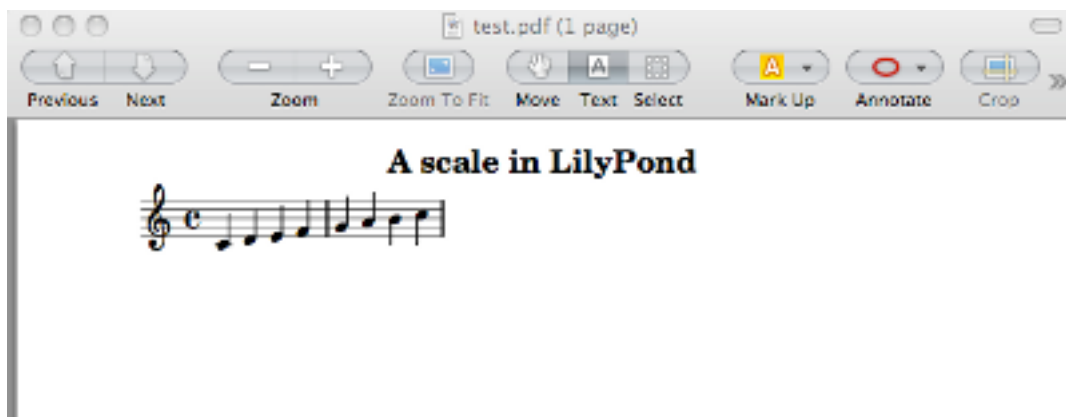


A new window will open showing a progress log of the compilation of the file you have just saved.



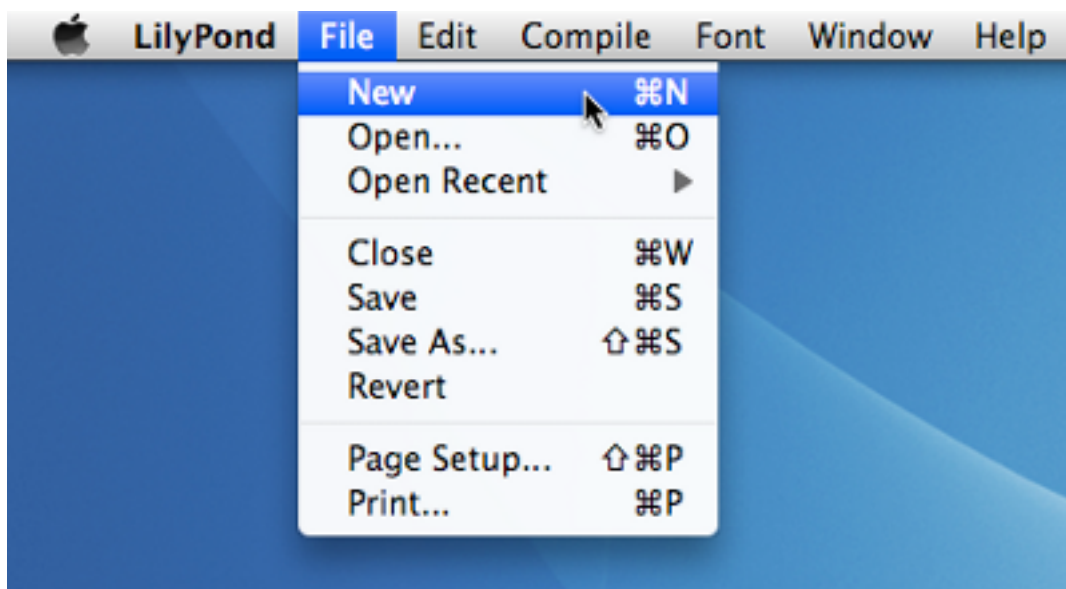
Step 3. View output

Once the compilation has finished, a PDF file will be created with the same name as the original file and will be automatically opened in the default PDF viewer and displayed on your screen.

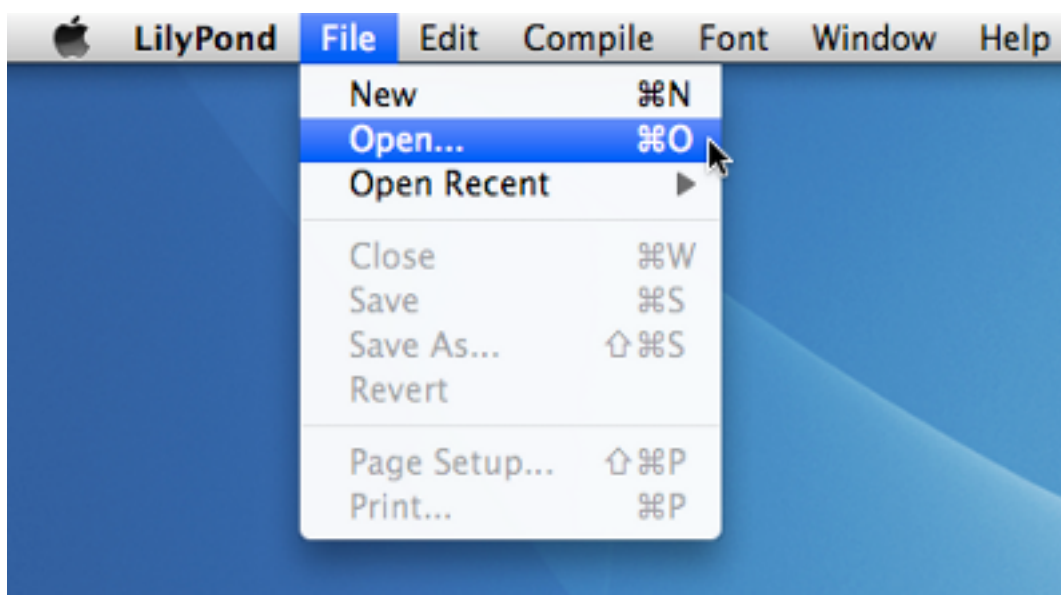


Other commands

To create new files for LilyPond, begin by selecting **File > New**



or **File > Open** to open and edit existing files you have saved previously.



You must save any new edits you make to your file before you **Compile > Typeset** and if the PDF file is not displayed check the window with the progress log for any errors.

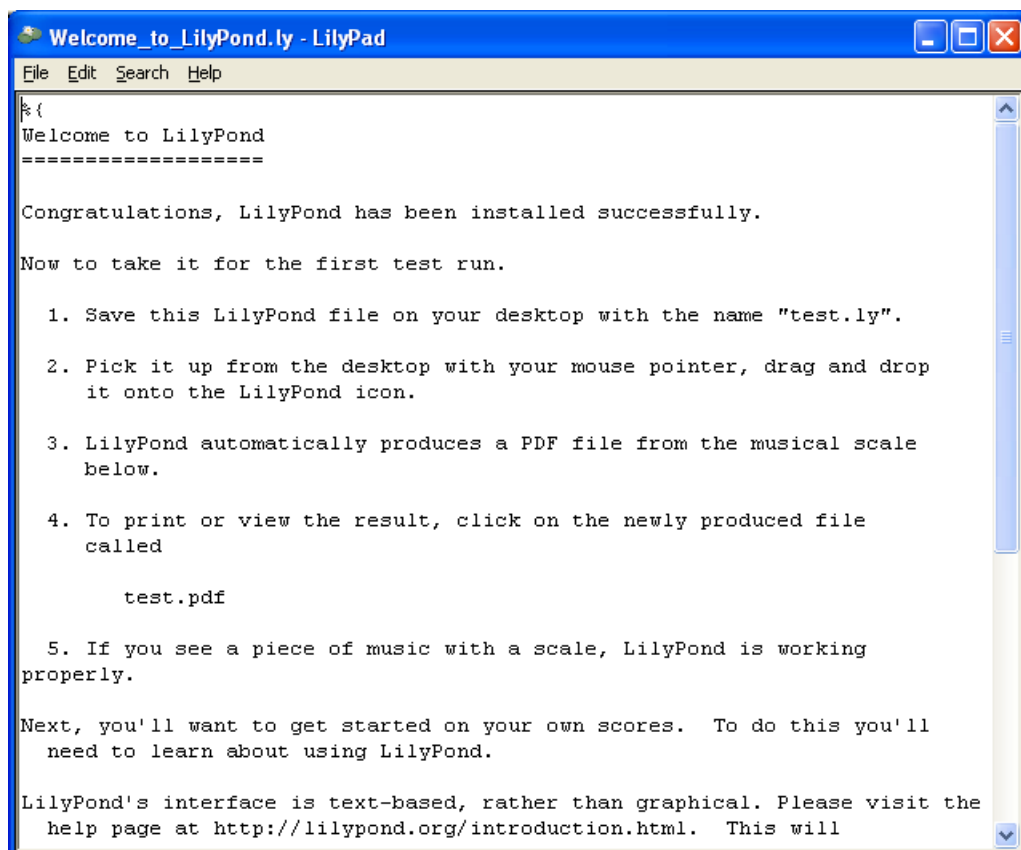
If you are not using the default Preview PDF viewer that comes with the Mac Operating system and you have the PDF file generated from a previous compilation open, then any further compilations may fail to generate an update PDF until you close the original.

1.1.3 Windows

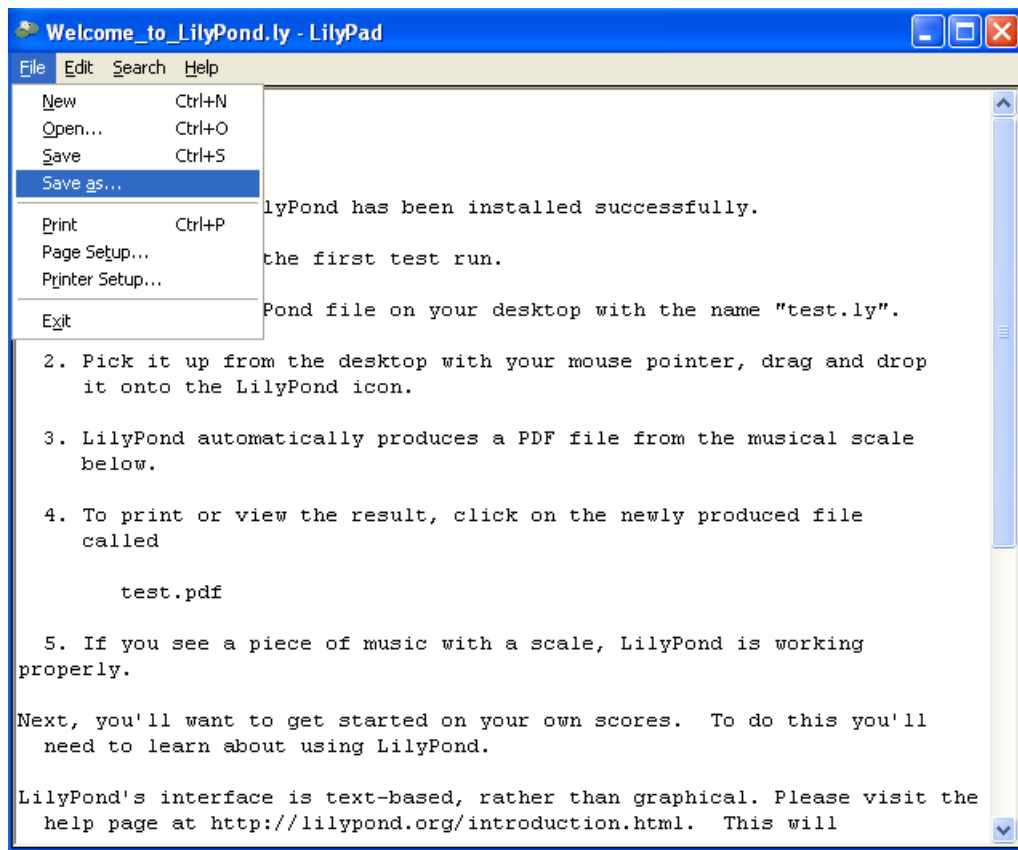
Nota: These instructions assume that you are using the built-in LilyPad editor. If you are using any of the programs described in *Secció “Easier editing” in Informació general*, consult the documentation for those programs should you have any problems.

Step 1. Create your '.ly' file

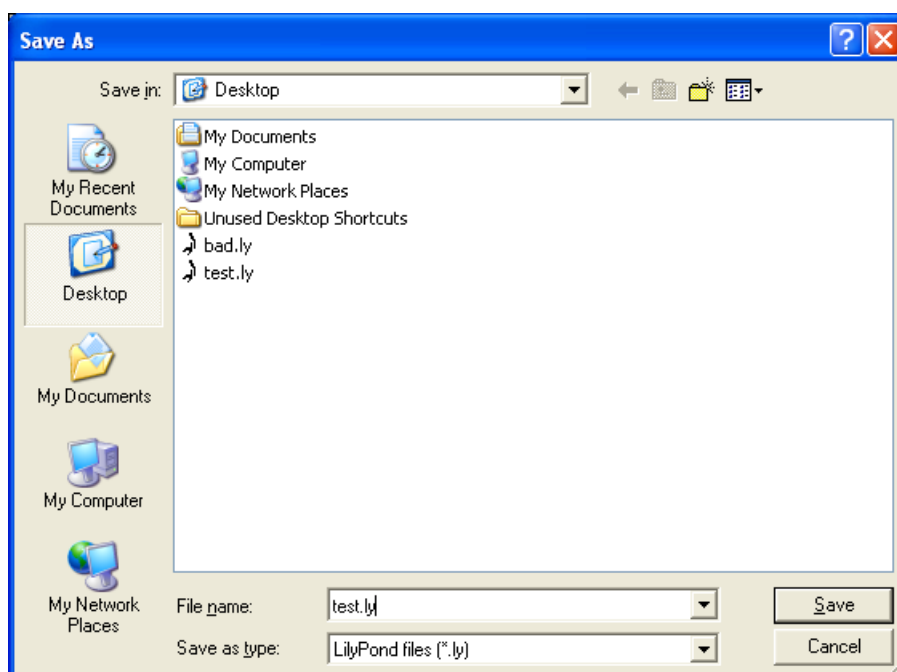
Double-click the LilyPond icon on your desktop and an example file will open.



From the menus that appear along the top of the example file, select **File > Save as**. Do not use the **File > Save** for the example file as this will not work until you have given it a valid LilyPond file name.



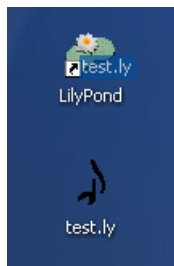
Choose a name for your file, for example 'test.ly'.



Step 2. Compile

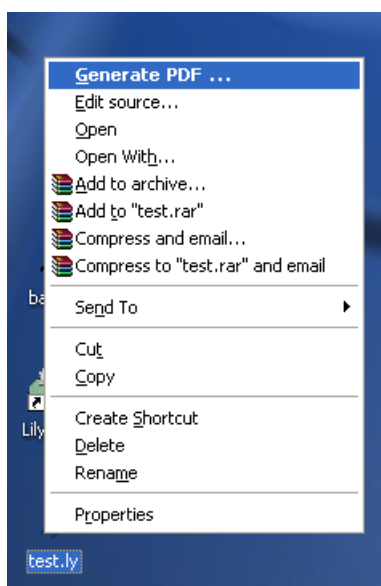
To turn your LilyPond file into a music score, you need to compile it. This can be done a number of ways – using drag and drop, with right-click, double-clicking or using the command line (a DOS box). We'll look at the first three to start with.

1. Drag-and-drop the file directly onto the LilyPond icon on the desktop.



Not much will seem to happen, but after a short while, you should see two new files on your desktop – ‘test.log’ and ‘test.pdf’.

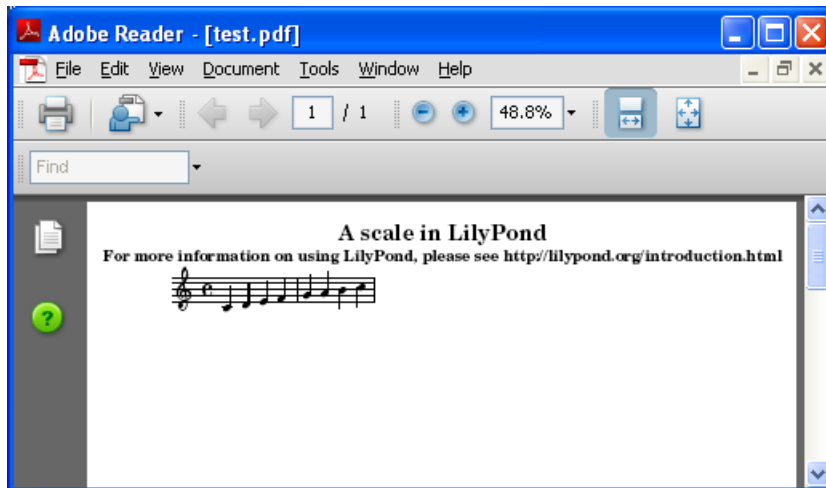
2. Right-click on the file and from the pop-up context menu and choose **Generate PDF**.



3. Or simply double-click the ‘test.ly’.

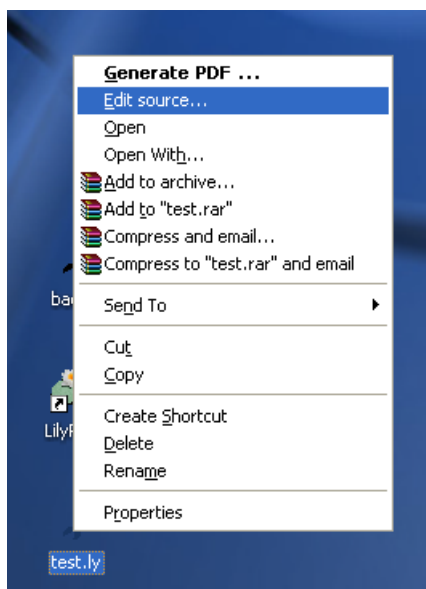
Step 3. View output

'test.pdf' contains the engraved 'test.ly' file. Double-click it and it should open in your PDF viewer:

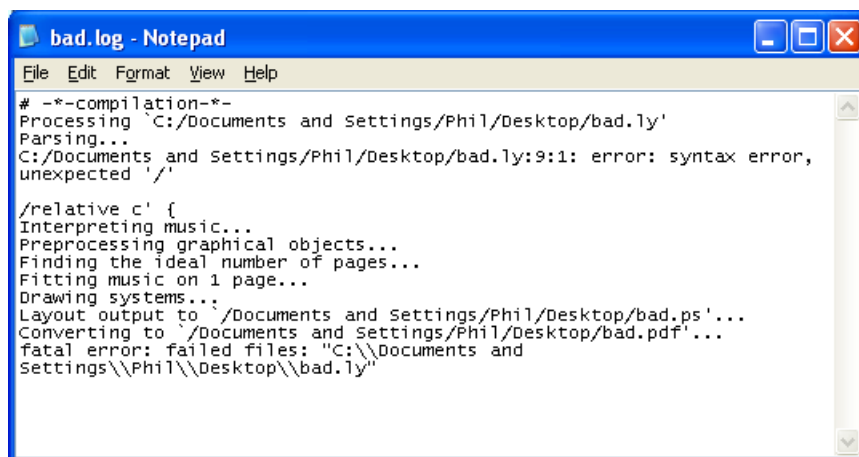


Other commands

To create a new file, begin by selecting **File > New** from within any previously created file or **File > Open** to open and edit any files you have saved before. You can also edit a file by right-clicking it and selecting **Edit source**.



You must save any edits you make before you try to compile your file. If the PDF file is not created or the output is not what you expected, check the log file that will have been created during the compilation attempt for any errors.



```

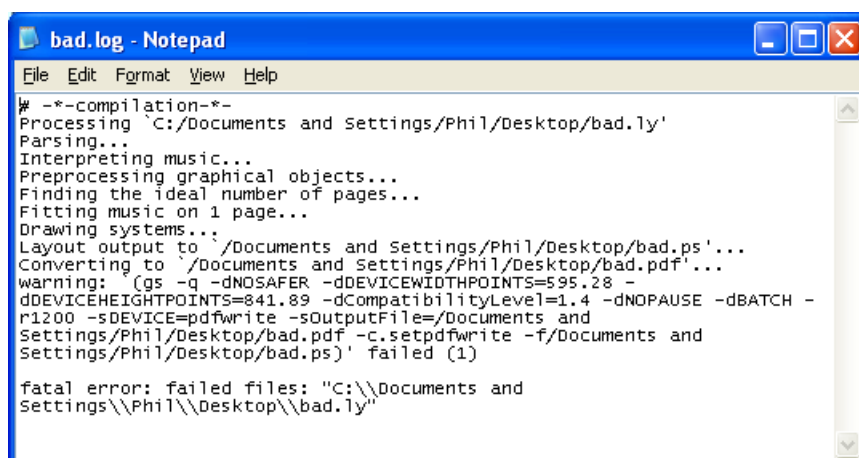
bad.log - Notepad
File Edit Format View Help
# -*-compilation-*-
Processing 'C:/Documents and Settings/Phil/Desktop/bad.ly'
Parsing...
C:/Documents and Settings/Phil/Desktop/bad.ly:9:1: error: syntax error,
unexpected '/'

/relative c' {
Interpreting music...
Preprocessing graphical objects...
Finding the ideal number of pages...
Fitting music on 1 page...
Drawing systems...
Layout output to 'C:/Documents and Settings/Phil/Desktop/bad.ps'...
Converting to 'C:/Documents and Settings/Phil/Desktop/bad.pdf'...
fatal error: failed files: "C:\\Documents and
Settings\\Phil\\Desktop\\bad.ly"

```

This log file is overwritten each time you compile your LilyPond file.

If you are viewing your file in a PDF viewer, then you must close the PDF if you wish to try a new compilation as it may fail to create the new PDF while it is still being viewed.



```

bad.log - Notepad
File Edit Format View Help
# -*-compilation-*-
Processing 'C:/Documents and Settings/Phil/Desktop/bad.ly'
Parsing...
Interpreting music...
Preprocessing graphical objects...
Finding the ideal number of pages...
Fitting music on 1 page...
Drawing systems...
Layout output to 'C:/Documents and Settings/Phil/Desktop/bad.ps'...
Converting to 'C:/Documents and Settings/Phil/Desktop/bad.pdf'...
warning: (gs -q -dNOSAfer -dDEVICEWIDTHPOINTS=595.28 -
dDEVICEHEIGHTPOINTS=841.89 -dCompatibilityLevel=1.4 -dNOPAUSE -dBATCH -
r1200 -sDEVICE=pdfwrite -sOutputFile=/Documents and
Settings/Phil/Desktop/bad.pdf -c.setpdfwrite -f/Documents and
Settings/Phil/Desktop/bad.ps) failed (1)

fatal error: failed files: "C:\\Documents and
Settings\\Phil\\Desktop\\bad.ly"

```

1.1.4 Línia d'ordres

Nota: These instructions assume that you are familiar with command-line programs. If you are using any of the programs described in [Secció “Easier editing” in Informació general](#), consult the documentation for those programs should you have any problems.

Step 1. Create your ‘.ly’ file

Create a text file called ‘test.ly’ and enter:

```

\version "2.18.2"
{
  c' e' g' e'
}

```


Step 2. Compile (with command-line)

To process ‘test.ly’, type the following at the command prompt:

```
lilypond test.ly
```

You will see something resembling:

```
GNU LilyPond 2.18.2
Processing `test.ly'
Parsing...
Interpreting music...
Preprocessing graphical objects...
Solving 1 page-breaking chunks...[1: 1 pages]
Drawing systems...
Layout output to `test.ps'...
Converting to `./test.pdf'...
Success: compilation successfully completed
```

Step 3. View output

You may view or print the resulting ‘test.pdf’.

1.2 Com escriure fitxers d’entrada

Aquesta secció presenta una part de la sintaxi bàsica del LilyPond com ajuda perquè us iniciieu a l’escriptura de fitxers d’entrada.

1.2.1 Notació senzilla

El LilyPond afegirà certs elements de notació de manera automàtica. A l’exemple següent hem especificat solament quatre altures, però el LilyPond ha afegit la clau, el compàs i les duracions.

```
{
  c' e' g' e'
}
```



Aquest comportament es pot modificar, però en general aquests valors automàtics són adequats.

Altures

Glossari musical: Secció “pitch” in *Glossari musical*, Secció “interval” in *Glossari musical*, Secció “scale” in *Glossari musical*, Secció “middle C” in *Glossari musical*, Secció “octave” in *Glossari musical*, Secció “accidental” in *Glossari musical*.

La manera més senzilla d’introduir les notes és mitjançant la utilització del model `\relative` (relatiu). En aquest mode, s’escull l’octava automàticament sota el supòsit que la següent nota es col·locarà sempre el més a prop de la nota actual, és a dir, es col·locarà a l’octava compresa dins de fins a tres espais de pentagrama a partir de la nota anterior. Començarem per introduir el fragment musical més elemental: una *escala*, on cada nota està compresa dins de tans sols un espai de pentagrama des de la nota anterior.

```
% set the starting point to middle C
\relative c' {
  c d e f
  g a b c
```

}



La nota inicial és el *Do central*. Cada nota successiva es col·loca el més a prop possible de la nota prèvia (en altres paraules: la primera 'c' és el Do més proper al Do central; a aquesta nota la segueix el Re més a prop a la nota prèvia, i així successivament). Podem crear melodies amb intervals més grans, fins i tot sense deixar d'utilitzar el mode relatiu:

```
\relative c' {
  d f a g
  c b f d
}
```



No és necessari que la primera nota de la melodia comenci exactament a la nota que especifica l'altura d'inici. A l'exemple anterior, la primera nota (d) és el Re més proper al Do central.

Afegint (o eliminant) cometes simples ' o comes , a l'ordre '\\relative c'', podem canviar l'octava d'inici:

```
% una octava per sobre del Do central
\\relative c'' {
  e c a c
}
```



Al principi, el mode relatiu pot resultar una mica confús, però és la forma més senzilla d'introduir la major part de les melodies. Vegem com funciona en la pràctica aquest càlcul relatiu. Començant per Si, que està situat a la línia central en clau de Sol, podem arribar Do, Re i Mi dins dels tres espais de pentagrama cap amunt, i La, Sol i Fa dins dels tres espais cap a baix. Per tant, si la nota següent a Si és Do, Re o Mi se suposarà que està per sobre del Si, mentre que La, Sol o Fa s'entendran situats per sota.

```
\\relative c'' {
  b c % el Do està 1 espai per sobre, és el Do de dalt
  b d % el Re està 2 espais per sobre o 5 per sota, és el Re de dalt
  b e % el Mi està 3 espais per sobre o 4 per sota, és el Mi de dalt
  b a % el La està 6 espais per sobre o 1 per sota, és el La de sota
  b g % el Sol està 5 espais per sobre o 2 per sota, és el Sol de sota
  b f % el Fa està 4 espais per sobre o 3 per sota, és el F de sota
}
```



El mateix exactament passa quan qualssevol d'aquestes notes porten un sostingut o un bemoll. Les *Alteracions accidentals* s' **ignoren totalment** al càlcul de la posició relativa. Exactament el mateix compte d'espais de pentagrama es fa a partir d'una nota situada en qualsevol altre lloc del mateix pentagrama.

Per afegir intervals més grans que tres espais de pentagrama, podem elevar una *octava* afegint una cometa simple ' (o apòstrof) a continuació del nom de la nota. També podem baixar una octava escrivint una coma , a continuació del nom de la nota.

```
\relative c'' {
  a a, c' f,
  g g'' a,, f'
}
```



Per pujar o baixar una nota en dos (o més!) octaves, utilitzem diverses '' o ,, (però teniu cura d'utilitzar dos cometes simples '' i no una cometa doble ") !)

Duracions (valors rítmics)

Glossari musical: Secció “beam” in *Glossari musical*, Secció “duration” in *Glossari musical*, Secció “whole note” in *Glossari musical*, Secció “half note” in *Glossari musical*, Secció “quarter note” in *Glossari musical*, Secció “dotted note” in *Glossari musical*.

La *duració* d'una nota s'especifica mitjançant un número després del nom de la nota: 1 significa *rodona*, 2 significa *blanca*, 4 significa *negra* i així successivament. Les *barres de corxera* s'afegeixen automàticament.

Si no especifiqueu una duració, s'utilitza la duració prèvia per a la nota següent. La figura per omissió de la primera nota és una negra.

```
\relative c'' {
  a1
  a2 a4 a8 a
  a16 a a a a32 a a a a64 a a a a a a a2
}
```



Per crear *notes amb puntet*, afegiu un punt . al número de la duració. La duració d'una nota amb puntet s'ha d'especificar de forma explícita (és a dir: mitjançant un número).

```
\relative c'' {
  a4 a a4. a8
  a8. a16 a a8. a8 a4.
}
```



Silencis

Glossari musical: *Secció “rest” in Glossari musical.*

Un *silenci* s'introdueix igual que si fos un anota amb el nom `r` :

```
\relative c'' {
  a4 r r2
  r8 a r4 r4. r8
}
```



Indicació de compàs

Glossari musical: *Secció “time signature” in Glossari musical.*

La *indicació de compàs* es pot establir amb l'ordre `\time` :

```
\relative c'' {
  \time 3/4
  a4 a a
  \time 6/8
  a4. a
  \time 4/4
  a4 a a a
}
```



Indicacions de tempo

Glossari musical: *Secció “tempo indication” in Glossari musical, Secció “metronome” in Glossari musical.*

La *indicació de tempo* i la *indicació de metrònom* poden establir-se amb l'ordre `\tempo`:

```
\relative c'' {
  \time 3/4
  \tempo "Andante"
  a4 a a
  \time 6/8
  \tempo 4. = 96
  a4. a
  \time 4/4
  \tempo "Presto" 4 = 120
  a4 a a a
}
```



Clau

Glossari musical: *Secció “clef” in Glossari musical.*

La *clau* es pot establir utilitzant l'ordre `\clef` :

```
\relative c' {
  \clef "treble"
  c1
  \clef "alto"
  c1
  \clef "tenor"
  c1
  \clef "bass"
  c1
}
```



Tot a l'hora

Aquí teniu un petit exemple que mostra tots els elements anteriors a l'hora:

```
\relative c, {
  \clef "bass"
  \time 3/4
  \tempo "Andante" 4 = 120
  c2 e8 c'
  g'2.
  f4 e d
  c4 c, r
}
```



Vegeu també

Referència de la notació: *Secció “Writing pitches” in Referència de la notació*, *Secció “Writing rhythms” in Referència de la notació*, *Secció “Writing rests” in Referència de la notació*, *Secció “Indicació de compàs” in Referència de la notació*, *Secció “Clau” in Referència de la notació.*

1.2.2 Treball sobre els fitxers d'entrada

Els fitxers d'entrada del LilyPond són com els fitxers font de molts llenguatges de programació corrents. Contenen un enunciat de versió, són sensibles a les majúscules i generalment els espais s'ignoren. Les expressions es formen amb claudàtors `{ }` i els comentaris es denoten per un signe de percentatge (%) o per `%{ ... }%`.

Si no enteneu res de la frase anterior, no us preocupeu! A continuació explicarem el significat de tots aquests termes:

- **Enunciat de la versió:** Tot fitxer del LilyPond ha de contenir un enunciat de versió. Un enunciat de versió és una línia que descriu la versió del LilyPond per a la es va escriure aquest fitxer, com a l'exemple següent:


```
comentari de bloc.

f4 f e e d d c2
%}
```

1.3 Gestió dels errors

A vegades el LilyPond no produeix el resultat esperat. Aquesta secció aporta alguns enllaços per ajudar-vos a resoldre els problemes que pugueu trobar.

1.3.1 Consells generals de solució de problemes

La solució de problemes al LilyPond pot ser un desafiament per a les persones acostumades als interfícies gràfics, perquè és possible crear fitxers d'entrada invàlids. Quan passa això, la millor manera d'identificar i resoldre el problema és aplicar un enfocament lògic. A [Secció “Solució de problemes” in *Utilització del programa*](#) es donen algunes guies per ajudar-vos a aprendre a fer-lo.

1.3.2 Alguns errors comuns

Hi ha alguns errors comuns que són difícils de solucionar si ens basem solament als missatges d'errors que se'ns presenten. Aquests errors es descriuen a [Secció “Errors comuns” in *Utilització del programa*](#).

1.4 Com llegir els manuals

Aquesta secció mostra com llegir la documentació de forma eficient, i presenta algunes funcionalitats interactives de la versió en línia.

1.4.1 Material omès

Com ja hem vist a [Secció 1.2.2 \[Treball sobre els fitxers d'entrada\]](#), [pàgina 16](#) codi d'entrada del LilyPond ha d'estar rodejat de claudàtors `{ }` o de `\relative c' { ... }`. Durant la resta del manual actual, la major part dels exemples ometran els claudàtors. Per a reproduir els exemples, haureu de copiar i enganxar l'entrada que es mostra, però **haureu** d'escriure el `\relative c' { ... }`, de la forma següent:

```
\relative c' {
  ...aquí va l'exemple...
}
```

Perquè ometre els claudàtors? Gairebé tots els exemples del manual actual es poden inserir al mig d'un fragment més gran de música. Per a aquests exemples no té cap sentit afegir `\relative c' { ... }` (no hauríeu de posar un `\relative` dins d'altre `\relative`!); si haguéssim inclòs `\relative c' { ... }` envoltant a cadascú dels exemples, no podríeu copiar un exemple petit procedent de la documentació i enganxar-lo dins de la seva pròpia partitura. La major part de la gent voldrà inserir el codi dins d'una partitura més gran, és per això que hem formatat el manual d'aquesta manera.

Recordeu també que tot fitxer del LilyPond ha de portar un enunciat `\version`. Considerant que els exemples dels manuals són fragments de codi i no fitxers complets, l'enunciat `\version` s'omet. Tot i així, ens hem d'acostumar a incloure'ls als nostres documents.

1.4.2 Exemples amb enllaç

Nota: Aquesta funcionalitat sols està disponible al manuals en HTML.

Moltes persones aprendran a utilitzar programes provant i trastejant amb ells. Això també es pot fer amb el LilyPond. Si cliqueu sobre una imatge a la versió en HTML d'aquest manual, podreu veure l'entrada exacta del LilyPond que es va fer servir per generar aquesta imatge. Proveu-lo sobre aquesta imatge:



Tallent i enganxant tot el que es troba dins de la secció “ly snippet” (fragment de tipus ly), tindreu una plantilla inicial per als vostres experiments. Per poder veure exactament el mateix resultat (amb la mateixa amplada i tot), copieu tot el que hi ha des de “Start cut-&-pastable section” fins al final del fitxer.

1.4.3 Panoràmica dels manuals

Hi ha molt abundant documentació sobre el LilyPond. Els nous usuaris es troben desorientats amb freqüència en quant a quina part o quines parts han de llegir, i ocasionalment passen per alt la lectura de parts d'importància vital.

Nota: Si us plau: no us salteu les parts importants de la documentació. Us resultarà molt més difícil comprendre les seccions subsegüents.

- **Abans d'intentar fer *qualsevol cosa*:** llegiu el [Capítol 1 \[Tutorial\]](#), [pàgina 1](#) del manual d'Aprenentatge, i la secció [Capítol 2 \[Notació corrent\]](#), [pàgina 20](#). Si trobeu termes musicals que no reconeixeu, busqueu-los al [Secció “Glossari” in *Glossari musical*](#).
- **Abans d'intentar escriure una peça completa de música:** llegiu la secció [Capítol 3 \[Conceptes fonamentals\]](#), [pàgina 41](#) del manual d'Aprenentatge . Després us vindrà bé consultar les seccions corresponents de la [Secció “Referència de la notació” in *Referència de la notació*](#).
- **Abans d'intentar de modificar els resultats predeterminats:** llegiu la secció [Capítol 4 \[Tweaking output\]](#), [pàgina 88](#) del manual d'Aprenentatge.
- **Abans d'afrontar un projecte gran:** llegiu la secció [Secció “Suggeriments per escriure fitxers d'entrada del LilyPond” in *Utilització del programa*](#) del manual d'utilització del programa.

2 Notació corrent

Aquest capítol explica com crear boniques partitures que continguin notació musical comú, com a continuació al material que està al [Capítol 1 \[Tutorial\]](#), [pàgina 1](#).

2.1 Notació en un sol pentagrama

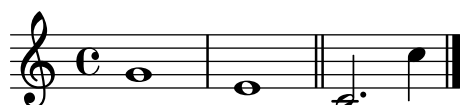
Aquesta secció presenta la notació comú que s'utilitza per a música a una veu sobre un pentagrama únic.

2.1.1 Línies divisòries i comprovacions de compàs

Línies divisòries

Les línies de compàs senzilles es dibuixen automàticament a la música, per la qual cosa no s'han d'afegir de forma manual. Altres tipus de barres de compàs s'afegeixen usant `\bar`, per exemple `\bar "||"` per a una doble barra, o `\bar "|."` per a la doble barra final. Per veure una llista completa de les línies divisòries, consulteu [Secció “Línies divisòries” in Referència de la notació](#).

```
g1 e1 \bar "||" c2. c'4 \bar "|."
```



Comprovacions de compàs

Tot i que no es estrictament necessari, és convenient utilitzar *comprovacions de compàs* dins del codi d'entrada per a indicar on se suposa que van les línies divisòries. S'introdueixen mitjançant el caràcter de barra vertical, `|` (AltGr -1 al teclat català). Amb les comprovacions de compàs, el programa pot verificar que hem introduït les duracions que completen els compassos correctament. Les comprovacions de compàs també fan que el codi d'entrada sigui més fàcil de llegir, perquè ajuden a organitzar el material.

```
g1 | e1 | c2. c'4 | g4 c g e | c4 r r2 |
```



Vegeu també

Referència de la notació: [Secció “Bar and bar number checks” in Referència de la notació](#).

2.1.2 Alteracions accidentals i armadures

Nota: Freqüentment els nous usuaris es confonen amb les alteracions accidentals: us preguem que llegeixi l'avertiment que apareix al final d'aquesta secció, especialment si no teniu familiaritat amb la teoria musical!

Alteracions accidentals

Glossari musical: Secció “sharp” in *Glossari musical*, Secció “flat” in *Glossari musical*, Secció “double sharp” in *Glossari musical*, Secció “double flat” in *Glossari musical*, Secció “accidental” in *Glossari musical*.

Una nota amb *sostingut* es fa afegint **is** al nom, i una nota *bemoll* afegint **es**. Como heu pogut endevinar, un *doble sostingut* o *doble bemol* es fa afegint **isis** o **eses**. Aquesta sintaxi deriva de les convencions de nomenclatura de les notes en les llengües nòrdiques i germàniques com l’alemany i l’holandès. Per utilitzar altres noms per a les *alteracions accidentals*, vegeu Secció “Note names in other languages” in *Referència de la notació*.

```
cis4 ees fisis, aeses
```



Armadures

Glossari musical: Secció “key signature” in *Glossari musical*, Secció “major” in *Glossari musical*, Secció “minor” in *Glossari musical*.

La *armadura de la tonalitat* se estableix mitjançant la instrucció `\key` seguit d’una nota i `\major` o `\minor`.

```
\key d \major
a1 |
\key c \minor
a1 |
```



Advertiment: armadures i altures

Glossari musical: Secció “accidental” in *Glossari musical*, Secció “key signature” in *Glossari musical*, Secció “pitch” in *Glossari musical*, Secció “flat” in *Glossari musical*, Secció “natural” in *Glossari musical*, Secció “sharp” in *Glossari musical*, Secció “transposition” in *Glossari musical*, Secció “Pitch names” in *Glossari musical*.

Per determinar si s’ha d’imprimir una notació *alteració accidental*, el LilyPond examina les notes i la *armadura de la tonalitat*. L’armadura solament afecta a les alteracions *impreses*, no a les pròpies notes! Aquesta funcionalitat sol confondre als que estan començant amb el programa, per això deixeu-nos explicar-la en detall.

El LilyPond fa una clara distinció entre el contingut musical i la presentació. L’alteració (*bemol*, *becaire* o *sostingut*) d’una nota és part de l’altura, i per tant és contingut musical. Si una alteració (un signe *impres* de bemoll, becaire o sostingut) s’imprimeix o no davant de la nota corresponent, és una qüestió de presentació. La presentació és quelcom que segueix unes regles, de manera que les alteracions accidentals s’imprimeixen automàticament segons aquestes regles. Les altures de les notes de la seva música són obres d’art, per tant no s’afegiran automàticament, i haureu d’introduir allò que vulgueu escoltar.

Al següent exemple:

```
\key d \major
cis4 d e fis
```



Cap nota porta una alteració impresa, però de totes maneres heu d'escriure l' `is a cis i a fis` al fitxers d'entrada.

El text `b` no significa “imprimir una boleta negra a la tercera línia del pentagrama.” El que significa en realitat: “hi ha una nota Si natural.” A la tonalitat de La bemoll major, *porta* una alteració accidental:

```
\key aes \major
aes4 c b c
```



Si l'anterior us causa confusió, penseu sobre el següent: si estiguéssiu tocant el piano, quina tecla polsàrieu? Si es tractés d'una tecla negra, aleshores *heu* d'afegir `-is` o `-es` al nom de la nota!

Posar totes les alteracions de forma explícita pot requerir una mica més de treball d'escriure, però l'avantatge és que la *transposició* és més fàcil, i les alteracions es poden imprimir seguint diverses convencions diferents. Consulteu [Secció “Automatic accidentals” in Referència de la notació](#) per veure exemples de com es poden imprimir alteracions d'acord amb regles diferents.

Vegeu també

Referència de la notació: [Secció “Note names in other languages” in Referència de la notació](#), [Secció “Alteracions accidentals” in Referència de la notació](#), [Secció “Automatic accidentals” in Referència de la notació](#), [Secció “Key signature” in Referència de la notació](#).

2.1.3 Lligadures d'unió i d'expressió

Lligadures d'unió

Glossari musical: [Secció “tie” in Glossari musical](#).

Una *lligadura d'unió* es crea adjuntant un caràcter d'accent `~` a la primera nota lligada:

```
g4~ 4 c2~ | 4~ 8 a~ 2 |
```



Lligadures d'expressió

Glossari musical: [Secció “slur” in Glossari musical](#).

Una *lligadura d'expressió* és una corba que es traça abastant diverses notes. Les notes inicial i final es marquen mitjançant `(` i `)` respectivament.

```
d4( c16) cis( d e c cis d) e( d4)
```



Lligadures de fraseig

Glossari musical: Secció “slur” in *Glossari musical*, Secció “phrasing” in *Glossari musical*.

Les lligadures que s'utilitzen per indicar *fraseigs* més llargs es poden introduir mitjançant \ (i \). Poden haver-hi a l'hora lligadures de legato i lligadures de fraseig, però no és possible tenir legatos simultanis o lligadures d'expressió simultànies.

g4\ (g8 (a) b (c) b4\)



Advertiments: lligadures d'expressió en front a lligadures d'unió

Glossari musical: Secció “articulation” in *Glossari musical*, Secció “slur” in *Glossari musical*, Secció “tie” in *Glossari musical*.

Una *lligadura d'expressió* sembla una *lligadura d'unió*, però té un significat diferent. Una lligadura (d'unió) senzillament feu que la primera nota sigui més llarga, i sols es pot utilitzar sobre parelles de notes iguals. Les lligadures d'expressió indiquen la *articulació* de les notes, i es poden utilitzar sobre grups majors de notes. Les lligadures d'unió i d'expressió es poden niuar unes a dins de les altres.

c4~ (c8 d~ 4 e)



Vegeu també

Referència de la notació: Secció “Lligadures d'unió” in *Referència de la notació*, Secció “Lligadures d'expressió” in *Referència de la notació*, Secció “Lligadures de fraseig” in *Referència de la notació*.

2.1.4 Articulations i matisos dinàmics

Articulacions

Glossari musical: Secció “articulation” in *Glossari musical*.

Les *articulacions* més corrents es poden afegir a les notes utilitzant un guió - seguit d'un caràcter únic:

c4-^ c-+ c-- c-!
c4-> c-. c2- _



Indicacions de digitació

Glossari musical: Secció “fingering” in *Glossari musical*.

De manera similar, les *digitacions* es poden afegir a una nota utilitzant un guió (-) seguit del dígit desitjat:

c4-3 e-5 b-2 a-1



Les articulacions i digitacions normalment es col·loquen de forma automàtica, però podeu especificar una direcció mitjançant `^` (a sobre) o `_` (a sota). També podeu usar diverses articulacions sobre la mateixa nota. No obstant, gairebé sempre és millor deixar que el LilyPond determini la direcció de les articulacions.

c4_-^1 d^ . f^4_2-> e^-_+



Matisos dinàmics

Glossari musical: Secció “dynamics” in *Glossari musical*, Secció “crescendo” in *Glossari musical*, Secció “decrescendo” in *Glossari musical*.

Les expressions de *matís* o signes dinàmics es fan afegint les marques (amb una barra invertida) a la nota:

c4\ff c\mf c\p c\pp



Els *crescendi* i *decrescendi* comencen amb les ordres `\<` i `\>`. La següent indicació de matís, com per exemple `\f`, acabarà el (de)crescendo, o bé es pot usar la instrucció `!`:

c4\< c\ff\> c c\!



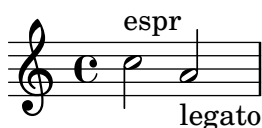
Vegeu també

Referència de la notació: Secció “Articulations and ornamentations” in *Referència de la notació*, Secció “Fingering instructions” in *Referència de la notació*, Secció “Matisos dinàmics” in *Referència de la notació*.

2.1.5 Addició de text

És possible afegir text a la partitura:

c2^"espr" a_"legato"



Es pot aplicar un format addicional mitjançant la instrucció `\markup`:

```
c2^\markup { \bold espr }
a2_\markup {
  \dynamic f \italic \small { 2nd } \hspace #0.1 \dynamic p
}
```



Vegeu també

Referència de la notació: Secció “Writing text” in *Referència de la notació*.

2.1.6 Barres automàtiques i manuals

Glossari musical: Secció “beam” in *Glossari musical*.

Totes les barres de les figures es dibuixen automàticament:

```
a8 ais d ees r d c16 b a8
```



Si no us agraden les barres automàtiques, poden forçar-se manualment. Marqueu la primera nota que comprèn la barra amb [i la última amb].

```
a8[ ais] d[ ees r d] c16 b a8
```



Si voleu desactivar completament el barrat automàtic o per a una secció extensa de música, utilitzeu l'ordre `\autoBeamOff` per desactivar-lo i `\autoBeamOn` para activar-lo un altre cop.

```
\autoBeamOff
a8 c b4 d8. c16 b4 |
\autoBeamOn
a8 c b4 d8. c16 b4 |
```



Vegeu també

Referència de la notació: Secció “Automatic beams” in *Referència de la notació*, Secció “Manual beams” in *Referència de la notació*.

2.1.7 Instruccions rítmiques avançades

Compàs parcial

Glossari musical: [Secció “anacrusis” in Glossari musical](#).

Una *anacrusi* s'introdueix amb la paraula clau `\partial`. Va seguida d'una duració: `\partial` 4 és una anacrusi de negra i `\partial` 8 de corxera.

```
\partial 8 f8 |
c2 d |
```



Grups especials

Glossari musical: [Secció “note value” in Glossari musical](#), [Secció “triplet” in Glossari musical](#).

Els grups especials com ara els tresets es fan amb la paraula clau `\tuplet`. Requereix dos arguments: una fracció i un fragment de música. La fracció és el nombre de notes del grup partit pel nombre de notes que normalment ocupen la mateixa duració. Per als tresets hi ha tres notes a l'espai de dues, de manera que els *tresets* es fan amb una fracció de 3/2.

```
\tuplet 3/2 { f8 g a }
\tuplet 3/2 { c8 r c }
\tuplet 3/2 { f,8 g16[ a g a] }
\tuplet 3/2 { d4 a8 }
```



Notes d'adorn

Glossari musical: [Secció “grace notes” in Glossari musical](#), [Secció “acciaccatura” in Glossari musical](#), [Secció “appoggiatura” in Glossari musical](#).

Les *notes d'adorn* es creen amb l'ordre `\grace`, tot i que també es poden aconseguir precedint una expressió musical amb la paraula clau `\appoggiatura` o `\acciaccatura`.

```
c2 \grace { a32 b } c2 |
c2 \appoggiatura b16 c2 |
c2 \acciaccatura b16 c2 |
```



Vegeu també

Referència de la notació: [Secció “Notes d'adorn” in Referència de la notació](#), [Secció “Grups especials” in Referència de la notació](#), [Secció “Upbeats” in Referència de la notació](#).

2.2 Diverses notes a la vegada

Aquesta secció és una introducció a les notes simultànies: diversos instruments, diversos pentagrames per a un sol instrument (per exemple piano) i acords.

La paraula “polifonia” en música fa referència al fet de tenir més d'una veu en un moment determinat dins d'una peça musical. La paraula “polifonia” al LilyPond es refereix al fet de tenir més d'una veu al mateix pentagrama.

2.2.1 Explicació de les expressions musicals

Als fitxers d'entrada del LilyPond, la música es representa mitjançant *expressions musicals*. Una sola nota és una expressió musical:

a4



En tancar un grup de notes dins de claudàtors creem una *expressió musical composta*. Aquí hem creat una expressió musical composta amb dues notes:

{ a4 g4 }



Si col·loquem un grup d'expressions musicals (per exemple: notes) dins de claudàtors, això significa que es troben en seqüència (és a dir, cada una segueix a l'anterior). El resultat és una altra expressió musical:

{ { a4 g } f4 g }



Analogia: expressions matemàtiques

Aquest mecanisme és semblant a les fórmules matemàtiques: una fórmula gran es construeix combinant fórmules petites. Aquestes fórmules es diuen expressions, i la seva definició és recursiva de tal manera que es poden construir expressions d'una mida i complexitat arbitràries. Per exemple:

1

1 + 2

(1 + 2) * 3

((1 + 2) * 3) / (4 * 5)

Això és una seqüència d'expressions on cada expressió es troba continguda dins de la següent, més gran. Les expressions més simples són números, i les majors es fan combinant expressions mitjançant operadors (coma ara +, * i /) i parèntesis. De la mateixa manera es pot niuar a una profunditat arbitrària, el que es fa necessari per a músiques complexes com ara les partitures polifòniques.

Expressions musicals simultànies: diversos pentagrames

Glossari musical: Secció “polyphony” in *Glossari musical*.

Aquesta tècnica és molt útil per a la música *polifònica*. Per introduir música amb més veus o amb més pentagrames, el que fem és combinar diverses expressions en paral·lel. Per indicar que dues veus s'han d'interpretar al mateix temps, senzillament introduïm una combinació simultània d'expressions musicals. Una expressió musical ‘simultània’ es forma tancant les expressions dins

de `<< y >>`. A l'exemple que segueix, tres seqüències (cadascuna de les quals conté dues notes diferents) es combinen de forma simultània:

```
\relative c'' {
  <<
    { a2 g }
    { f2 e }
    { d2 b }
  >>
}
```



Tingueu en compte que hem sagnat cada nivell jeràrquic de l'entrada amb un marge diferent. Al LilyPond no li importa quant (o que poc) espai hi hagi al començament d'una línia però l'establiment de marges diferents dins del codi del LilyPond, d'aquesta forma, el fa molt més fàcil de llegir per a nosaltres els éssers humans.

Nota: Cada nota s'entén relativa a la nota anterior de l'entrada, solament la primera és relativa a la `c''` dins de la instrucció inicial `\relative`.

Expressions musicals simultànies: un sol pentagrama

Per determinar el nombre de pentagrames a una peça, el LilyPond examina el començament de la primera expressió. Si hi ha una sola nota, hi ha un sol pentagrama; si hi ha una expressió simultània, hi ha més d'un pentagrama. El següent exemple presenta una expressió complexa, però com comença amb una sola nota, es disposa sobre un sol pentagrama.

```
\relative c'' {
  c2 <<c e>> |
  << { e2 f } { c2 <<b d>> } >> |
}
```



2.2.2 Diversos pentagrames

Com ja hem vist a [Secció 2.2.1 \[Explicació de les expressions musicals\]](#), pàgina 27, els fitxers d'entrada per al LilyPond es construeixen a base d'expressions musicals. Si la partitura comença amb expressions musicals simultànies, el LilyPond crea diversos pentagrames. És més fàcil, no obstant, veure el que passa si creem cada u dels pentagrames de forma explícita.

Per imprimir més d'un pentagrama, cada fragment de música que constitueix un pentagrama es marca escrivint `\new Staff` abans d'ell. Aquests elements `Staff` es combinen després en paral·lel amb `<< y >>`:

```
\relative c'' {
  <<
    \new Staff { \clef "treble" c4 }
    \new Staff { \clef "bass" c,,4 }
  >>
}
```



L'ordre `\new` inaugura un 'context de notació'. Un context de notació és un entorn dins del què s'interpreten els esdeveniments musicals (com les notes o les ordres `\clef`). Per peces senzilles, els contextos d'aquest tipus es creen automàticament. Per a peces més complicades, és millor marcar els contextos de forma explícita.

Hi ha diverses classes de contextos. `Score`, `Staff` i `Voice` gestionen la notació melòdica, mentre que `Lyrics` s'ocupa dels textos cantats i `ChordNames` imprimeix els noms dels acords.

En termes de sintaxi, l'anteposició de `\new` a una expressió musical crea una expressió musical major. És semblant al signe menys de les matemàtiques. La fórmula $(4 + 5)$ és una expressió, per tant $-(4 + 5)$ és una expressió més àmplia.

Les indicacions de compàs escrites a un pentagrama afecten la resta d'ells, de forma predeterminada. En canvi, l'armadura de la tonalitat d'un pentagrama *no* afecta als altres pentagrames. Aquest comportament predeterminat diferent és a causa de què les partitures amb instruments transpositors són més comunes que les partitures polirítmiques.

```
\relative c'' {
  <<
    \new Staff { \clef "treble" \key d \major \time 3/4 c4 }
    \new Staff { \clef "bass" c,,4 }
  >>
}
```



2.2.3 Grups de pentagrames

Glossari musical: *Secció “brace” in Glossari musical*, *Secció “staff” in Glossari musical*, *Secció “system” in Glossari musical*.

La música per a piano es compon tipogràficament en forma de dos pentagrames units mitjançant un *claudàtor*. L'aspecte imprès d'aquest sistema de pentagrames se sembla a l'exemple polifònic que apareix a *Secció 2.2.2 [Diversos pentagrames], pàgina 28*, però en aquest cas l'expressió completa es col·loca dins d'un `PianoStaff`:

```
\new PianoStaff <<
  \new Staff ...
  \new Staff ...
>>
```

Heus aquí un petit exemple:

```
\relative c'' {
  \new PianoStaff <<
    \new Staff { \time 2/4 c4 e | g g, | }
    \new Staff { \clef "bass" c,,4 c' | e c | }
  >>
}
```



Altres grups de pentagrames es declaren mitjançant `\new GrandStaff`, que és apropiat per a partitures orquestrals, i `\new ChoirStaff`, que és apropiat per a partitures vocals. Cadascú d'aquests grups de pautes forma un tipus de context diferent, que produeix el claudàtor a l'esquerra i que també controla l'abast de les línies divisòries.

Vegeu també

Referència de la notació: Secció “Keyboard and other multi-staff instruments” in *Referència de la notació*, Secció “Displaying staves” in *Referència de la notació*.

2.2.4 Combinar notes per formar acords

Glossari musical: Secció “chord” in *Glossari musical*.

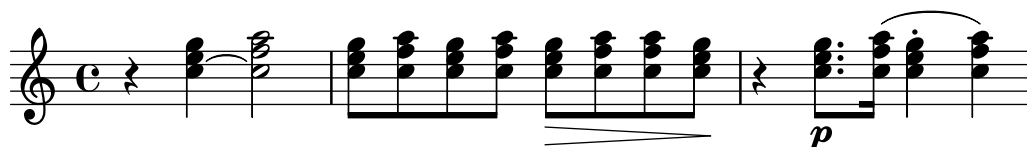
Hem vist amb anterioritat com es poden combinar les notes formant *acordes* que indiquen que són simultànies, tancant-les entre dobles angles. La forma normal d'indicar un acord, però, és tancar les notes entre angles *senzills*. Observeu que totes les notes d'un acord han de tenir la mateixa duració, i que la duració s'escriu després de l'angle de tancament.

```
r4 < c e g> < c f a>2
```



Hem de pensar que els acords són quelcom gairebé equivalent a les notes senzilles: gairebé tot el es pot adjuntar a una nota es pot adjuntar també a un acord, i tot ha d'anar *per fora* dels angles. Per exemple, podeu combinar marques com ara barres i lligadures, amb acords. Tan sols heu de recordar que s'escriuen per fora dels angles.

```
r4 < c e g>~ < c f a>2 |
< c e g>8[ < c f a> < c e g> < c f a>]
< c e g>8\>[ < c f a> < c f a> < c e g>]\! |
r4 < c e g>8.\p < c f a>16( < c e g>4-. < c f a>) |
```



Vegeu també

Referència de la notació: [Secció “Chorded notes” in Referència de la notació.](#)

2.2.5 Polifonia a un sol pentagrama

La música polifònica al LilyPond, tot i que no és difícil, utilitza conceptes que encara no hem tractat, per la qual cosa no les presentarem en aquest moment. En comptes d'això, les seccions següents presenten aquests conceptes i els expliquen en profunditat.

Vegeu també

Manual d'aprenentatge: [Secció 3.2 \[Les veus contenen música\], pàgina 48.](#)

Referència de la notació: [Secció “Simultaneous notes” in Referència de la notació.](#)

2.3 Cançons

En aquesta secció presentem com elaborar música vocal i fulls senzilles de cançó.

2.3.1 Elaborar cançons senzilles

Glossari musical: [Secció “lyrics” in Glossari musical.](#)

Presentem a continuació l'inici de la melodia d'una cançó infantil, “Girls and boys come out to play”:

```
\relative c'' {
  \key g \major
  \time 6/8
  d4 b8 c4 a8 | d4 b8 g4
}
```



Es pot assignar la *lletra* a aquestes notes combinant ambdues amb la paraula clau `\addlyrics`. La lletra s'escriu separant cada síl·laba mitjançant un espai.

```
<<
  \relative c'' {
    \key g \major
    \time 6/8
    d4 b8 c4 a8 | d4 b8 g4
  }
  \addlyrics {
    Girls and boys come | out to play,
  }
>>
```



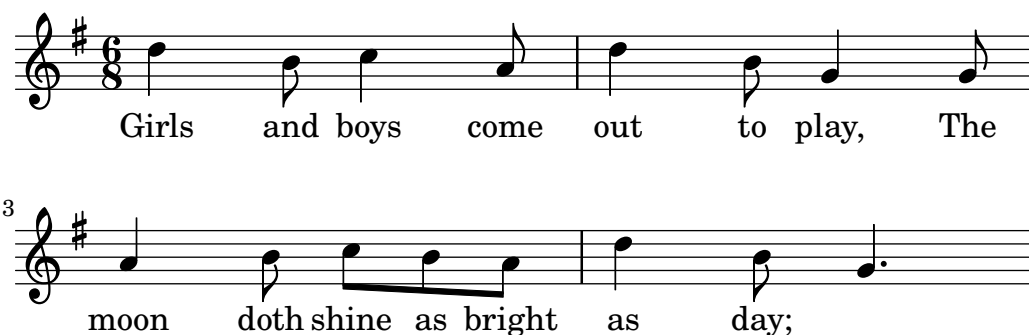
Observeu els angles dobles `<<...>>` al voltant del fragment sencer per expressar que la música i la lletra han de succeir al mateix temps.

2.3.2 Alineació de la lletra a una melodia

Glossari musical: Secció “melisma” in *Glossari musical*, Secció “extender line” in *Glossari musical*.

La següent línia de la cançó infantil és *The moon doth shine as bright as day*. A continuació l'ampliarem:

```
<<
\relative c'' {
  \key g \major
  \time 6/8
  d4 b8 c4 a8 | d4 b8 g4 g8 |
  a4 b8 c b a | d4 b8 g4. |
}
\addlyrics {
  Girls and boys come | out to play,
  The | moon doth shine as | bright as day; |
}
>>
```



Si compilem el codi de l'exemple anterior, veurem alguns missatges d'avertiment sobre la consola:

```
song.ly:12:29: warning: la comprovació de compàs ha fallat a: 5/8
The | moon doth shine as
                        | bright as day; |
song.ly:12:46: warning: la comprovació de compàs ha fallat a: 3/8
The | moon doth shine as | bright as day;
|
```

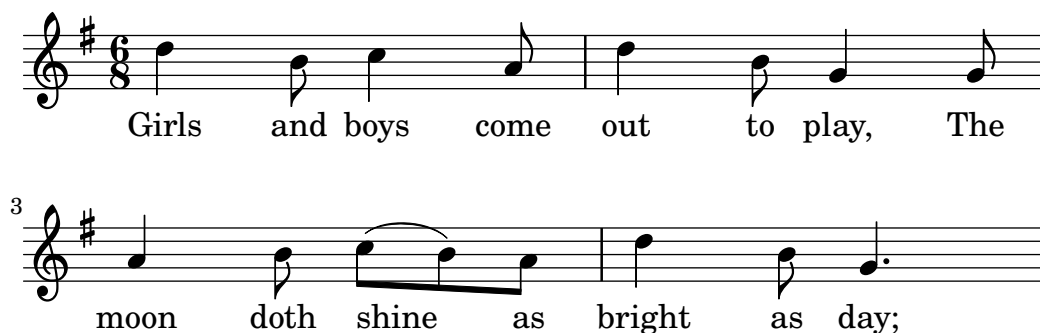
Aquest és un bon exemple de la utilitat de les comprovacions de compàs. Ara bé, si mirem la música, podem observar que la lletra addicional no s'alinea correctament amb les notes. La paraula ‘shine’ s’ha de cantar sobre dues notes, no una. Això es coneix com *melisma*, una síl·laba única que es canta sobre més d’una nota. Existeixen diverses formes de fer que una síl·laba recaigui sobre diverses notes, sent la més senzilla escriure una lligadura d’expressió sobre elles (vegeu [Secció 2.1.3 \[Lligadures d’unió i d’expressió\]](#), pàgina 22):

```
<<
\relative c'' {
  \key g \major
  \time 6/8
  d4 b8 c4 a8 | d4 b8 g4 g8 |
  a4 b8 c( b) a | d4 b8 g4. |
}
\addlyrics {
```

```

Girls and boys come | out to play,
The | moon doth shine as | bright as day; |
}
>>

```

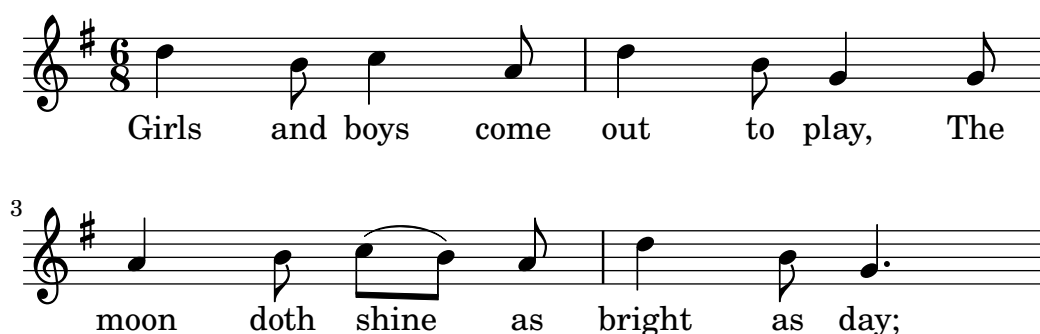


Ara la lletra s'alinea correctament amb les notes, però el barrat automàtic de les notes que corresponen a *shine as* no sembla correcte. Podem remeiar-lo inserint instruccions de barrat manual per sobreesciure el barrat automàtic; per veure més detalls consulteu [Secció 2.1.6 \[Barres automàtiques i manuals\]](#), pàgina 25.

```

<<
\relative c'' {
  \key g \major
  \time 6/8
  d4 b8 c4 a8 | d4 b8 g4 g8 |
  a4 b8 c([ b]) a | d4 b8 g4. |
}
\addlyrics {
  Girls and boys come | out to play,
  The | moon doth shine as | bright as day; |
}
>>

```



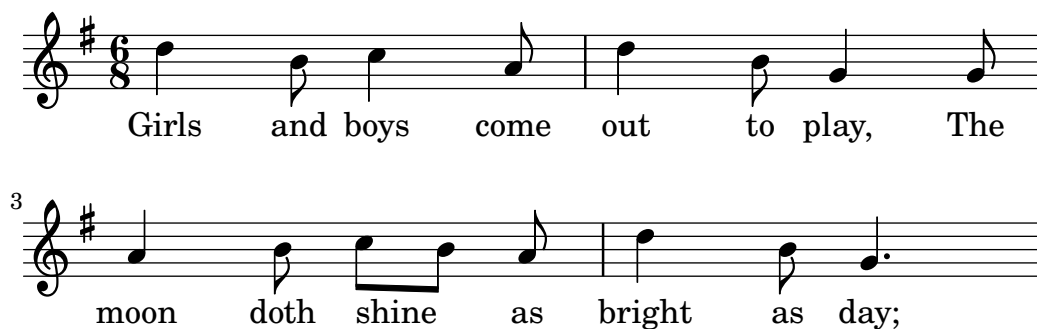
Com alternativa a la utilització de lligadures d'expressió, els melismes es poden indicar solament a la lletra utilitzant un guió baix, `_`, per a cada nota que volem incloure dins del melisma:

```

<<
\relative c'' {
  \key g \major
  \time 6/8
  d4 b8 c4 a8 | d4 b8 g4 g8 |
  a4 b8 c[ b] a | d4 b8 g4. |
}

```

```
\addlyrics {
  Girls and boys come | out to play,
  The | moon doth shine _ as | bright as day; |
}
>>
```



Si una síl·laba s'estén sobre diverses notes o una sola nota molt llarga, normalment es traça una *línia extensora* des de la síl·laba que s'estén i per sota de totes les notes que corresponen a aquesta síl·laba. S'escriu com dos guions baixos __. Heus aquí un exemple extret dels primers tres compassos del *Lament de Dido*, de *Dido i Enees* de Purcell:

```
<<
\relative c'' {
  \key g \minor
  \time 3/2
  g2 a bes | bes2( a) b2 |
  c4.( bes8 a4. g8 fis4.) g8 | fis1
}
\addlyrics {
  When I am | laid,
  am | laid __ in | earth,
}
>>
```



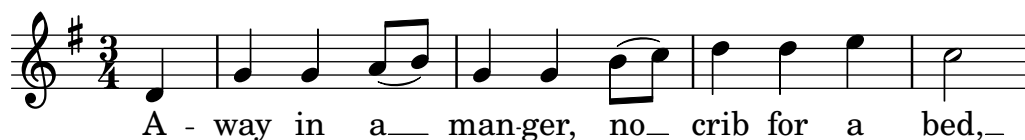
Fins al moment, cap dels exemples implicaven paraules que tinguessin més d'una síl·laba. Aquestes paraules es reparteixen en general a raó d'una nota per cada síl·laba, amb guions curts entre les síl·labes. Aquests guions separadors es teclegen com dos guions, amb el resultat d'un guió curt centrat entre les síl·labes. Presentem a continuació un exemple que demostra això i tot el que hem après fins aquest moment sobre l'alineació de la lletra a les notes.

```
<<
\relative c' {
  \key g \major
  \time 3/4
  \partial 4
  d4 | g4 g a8( b) | g4 g b8( c) |
  d4 d e | c2
}
\addlyrics {
```

```

A -- | way in a __ | man -- ger,
no __ | crib for a | bed, __
}
>>

```

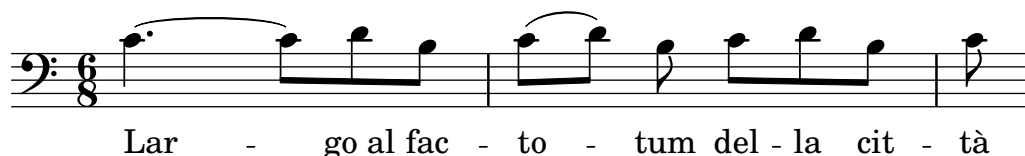


Alguns textos (especialment els que estan en italià o català) requereixen el contrari: col·locar més d'una síl·laba a una única nota. Això s'aconsegueix enllaçant les síl·labes entre sí mitjançant un guió baix simple _ (sense cap espai), o bé envoltant-les entre claudàtors. Aquí apareix un exemple procedent del *Barber de Sevilla* de Rossini, on la síl·laba *al* es canta sobre la mateixa nota que la síl·laba *go* de la paraula 'Largo' a l'ària de Fígaro *Largo al factotum*:

```

<<
\relative c' {
  \clef "bass"
  \key c \major
  \time 6/8
  c4.~ 8 d b | c8([ d]) b c d b | c8
}
\addlyrics {
  Lar -- go_al fac -- | to -- tum del -- la cit -- | tà
}
>>

```



Vegeu també

Referència de la notació: [Secció “Vocal music” in Referència de la notació.](#)

2.3.3 Lletra en diversos pentagrames

La solució senzilla que utilitza `\addlyrics` es pot usar per posar lletra a més d'un pentagrama. Aquí apareix un exemple tret del *Judas Macabeu* de Haendel:

```

<<
\relative c'' {
  \key f \major
  \time 6/8
  \partial 8
  c8 | c8([ bes]) a a([ g]) f | f'4. b, | c4.~ 4
}
\addlyrics {
  Let | flee -- cy flocks the | hills a -- | dorn, __
}
\relative c' {
  \key f \major

```



```

\time 6/8
\partial 8
r8 | r4. r4 c8 | a'8([ g]) f f([ e]) d | e8([ d]) c bes'4
}
\addlyrics {
  Let | flee -- cy flocks the | hills a -- dorn,
}
>>

```



Qualsevol partitura d'una complexitat més gran que la d'aquest senzill exemple es fa millor separant la lletra de l'estructura de pentagrames mitjançant variables (expressions amb nom). Les variables es tracten a [Secció 2.4.1 \[Organitzar les peces mitjançant variables\]](#), pàgina 36.

Vegeu també

Referència de la notació: [Secció “Vocal music” in Referència de la notació.](#)

2.4 Retocs finals

Aquest és l'últim apartat del tutorial; mostra la forma de donar els retocs finals a peces senzilles, i ofereix una introducció a la resta del manual.

2.4.1 Organitzar les peces mitjançant variables

Quan els elements que hem discutit anteriorment es combinen per produir fitxers més grans, les expressions musicals es fan enormes. A la música polifònica amb molts pentagrames, els fitxers d'entrada poden tornar-se molt propensos a la confusió, Podem reduir aquesta confusió utilitzant les *variables*.

Amb les variables (també conegudes com identificadors o macros), podem trossejar les expressions musicals complexes. Una variable s'assigna de la manera següent:

```
musicaAmbNom = { ... }
```

El contingut de l'expressió musical `musicaAmbNom` es pot usar posteriorment col·locant una barra invertida davant del nom (`\musicaAmbNom`, igual que una ordre normal del LilyPond).

```

violin = \new Staff {
  \relative c'' {
    a4 b c b
  }
}

```

```

cello = \new Staff {
  \relative c {
    \clef "bass"
    e2 d
  }
}

```

```
{
  <<
    \violin
    \cello
  >>
}
```



El nom d'una variable ha de consistir enterament de caràcters alfabètics, és a dir sense números, guions, ni guions baixos.

Les variables s'han de definir *abans* de l'expressió musical principal, però es poden usar tantes vegades com es vulgui, en qualsevol lloc, un cop que han estat definides. Fins i tot es poden usar dins de la definició d'una altra variable, proporcionant una via per escurçar el codi si una secció musical es repeteix moltes vegades.

```
tresilloA = \tuplet 3/2 { c,8 e g }
compasA = { \tresilloA \tresilloA \tresilloA \tresilloA }

\relative c'' {
  \compasA \compasA
}
```



Les variables es poden usar per a molts altres tipus d'objectes dins del codi d'entrada. Per exemple,

```
ancho = 4.5\cm
nombre = "Wendy"
papelAcinco = \paper { paperheight = 21.0 \cm }
```

Depenent del seu contingut, la variable es pot usar en diferents llocs. El següent exemple utilitza les variables anteriors:

```
\paper {
  \papelAcinc
  line-width = \ample
}

{
  c4^\nom
}
```

2.4.2 Afegir títols

La informació sobre el títol, autor, número d'Opus i altres elements similars s'escriuen al bloc `\header`. Aquest bloc es troba fora de l'expressió musical principal: el bloc `\header` normalment s'ubica per sota del número de versió.

```
\version "2.19.16"

\header {
  title = "Sinfonia"
  composer = "Jo"
  opus = "Op. 9"
}

{
  ... música ...
}
```

Quan es processa el fitxer, el títol i l'autor s'imprimeixen a sobre de la música. Podeu obtenir més informació sobre els títols a [Secció “Creating titles headers and footers” in Referència de la notació](#).

2.4.3 Noms de nota absoluts

Fins al moment sempre hem utilitzat `\relative` per definir les altures. Aquesta és normalment la forma més ràpida d'escriure la major part de la música. Sense `\relative`, les altures s'interpreten en mode absolut.

En aquest mode, el LilyPond tractarà totes les altures com valors absoluts. Una `c'` significarà sempre un Do central, una `b` significarà sempre la nota immediatament per sota del Do central i una `g`, significarà sempre la nota que es col·loca a la primera línia del pentagrama en clau de Fa.

```
{
  \clef "bass"
  c'4 b g, g, |
  g,4 f, f c' |
}
```



Heus aquí una escala que abasta quatre octaves:

```
{
  \clef "bass"
  c,4 d, e, f, |
  g,4 a, b, c |
  d4 e f g |
  a4 b c' d' |
  \clef "treble"
  e'4 f' g' a' |
  b'4 c'' d'' e'' |
  f''4 g'' a'' b'' |
  c''''1 |
}
```



Com podeu veure, escriure una melodia en clau de Sol implica escriure una gran quantitat d'apòstrofs " ". Considerem aquest fragment de Mozart:

```
{
  \key a \major
  \time 6/8
  cis''8. d''16 cis''8 e''4 e''8 |
  b'8. cis''16 b'8 d''4 d''8 |
}
```



Tots aquests apòstrofs fan gairebé illegible el codi d'entrada i serà origen de nombrosos errors. Amb `\relative`, l'exemple anterior és molt més fàcil de llegir:

```
\relative c' {
  \key a \major
  \time 6/8
  cis8. d16 cis8 e4 e8 |
  b8. cis16 b8 d4 d8 |
}
```



Si feu un error amb una marca d'octava (' o ,) mentre treballeu al mode `\relative`, serà molt obvi (moltes notes estaran a l'octava equivocada). Mentre treballeu en mode absolut, una sola fallada no serà tan visible, i tampoc no serà tan fàcil de localitzar.

El mode absolut és útil però per escriure música que contingui intervals grans, i serà extremadament útil per fer fitxers del LilyPond generats per ordinador. Quan es copien i peguen fragments melòdics, el mode absolut preserva l'octava del material original.

A vegades la música es disposa en formes més complexes. Si esteu usant `\relative` dins de `\relative`, les seccions relatives externa i interna són independents:

```
\relative c { c'4 \relative c' { f g } c }
```



Si esteu usant música absoluta dins d'un bloc de música relativa, tindreu que marcar la música absoluta explícitament amb la instrucció `\absolute` per evitar que s'incorpori a la música relativa:

```
\relative c { c'4 \absolute { f' g' } c }
```



2.4.4 Més enllà del tutorial

Després d'acabar el tutorial, potser hauríeu de provar a escriure una o dues peces. Comenceu amb una de les plantilles que apareixen a [Secció “Plantilles” in *Manual d'aprenentatge*](#) i afegiu algunes notes. Si necessiteu un tipus de notació que no ha estat tractat en aquest tutorial, doneu una ullada a la Referència de Notació, començant per [Secció “Musical notation” in *Referència de la notació*](#). Si voleu escriure música per a un conjunt instrumental que no estigui cobert per cap plantilla, consulteu [Secció 3.4 \[Extensió de les plantilles\], pàgina 70](#).

Un cop que heu escrit algunes peces curtes, llegiu la resta del Manual d'aprenentatge (capítols 3 al 5). Per suposat no passa res per llegir-lo ara mateix! La resta del Manual d'Aprenentatge, però, dona per suposat que teniu familiaritat amb l'entrada del LilyPond. Podeu saltar-vos aquests capítols ara i tornar a ells quan hagueu adquirit més experiència.

En aquest tutorial i a la resta del Manual d'Aprenentatge, hi ha un apartat **Vegeu també** al final de cada una de les seccions, que conté referències creuades a altres seccions: no seguiu aquestes referències durant la primera lectura: quan hagueu llegit el Manual d'Aprenentatge complet, potser desitgeu rellegir certes seccions i seguir les referències creuades per obtenir més informació.

Si no ho heu fet ja, us *preguem* que llegiu [Secció 1.4.3 \[Panoràmica dels manuals\], pàgina 19](#). Hi ha una gran quantitat d'informació sobre el LilyPond, de manera que els novinguts amb freqüència no saben exactament on han de buscar l'ajuda. Si dediqueu cinc minuts a llegir curosament aquesta secció us estalviareu hores de frustració buscant al lloc equivocat!

3 Conceptes fonamentals

Heu pogut veure al tutorial com produir música impresa amb bellesa a partir d'un simple fitxer de text. Aquesta secció presenta els conceptes i tècniques que es requereixen per produir partitures igualment belles però més complexes.

3.1 Com funcionen els fitxers d'entrada del LilyPond

El format d'entrada del LilyPond és força lliure en la seva forma i concedeix els usuaris amb experiència molta flexibilitat per estructurar els seus fitxers de la forma desitjada. Nogensmenys, tota aquesta flexibilitat pot fer que les coses es tornin confuses per als nous usuaris. Aquesta secció us explicarà part d'aquesta estructura, però pot obviar certs detalls per simplificar. Per veure una descripció més completa del format d'entrada, consulteu [Secció “File structure” in Referència de la notació](#).

3.1.1 Introducció a l'estructura dels fitxers del LilyPond

Un exemple bàsic de fitxer d'entrada del LilyPond és el següent:

```
\version "2.19.16"

\header { }

\score {
  ... expressió musical composta ... % tota la música va aquí
  \layout { }
  \midi { }
}
```

Hi ha moltes variacions d'aquest esquema bàsic, però l'exemple constitueix un punt de partida útil.

Fins al moment, cap dels exemples que heu pogut veure utilitza la instrucció `\score{}`. Això és així perquè el LilyPond afegeix automàticament les ordres addicionals que es requereixen quan li proporcionem una entrada senzilla. El LilyPond tracta una entrada com aquesta:

```
\relative c' {
  c4 a d c
}
```

com una abreviatura d'aquesta altra:

```
\book {
  \score {
    \new Staff {
      \new Voice {
        \relative c' {
          c4 a b c
        }
      }
    }
  }
  \layout { }
}
```

En altres paraules, si l'entrada consta d'una única expressió musical, el LilyPond interpreta el fitxer com si l'expressió musical estigués rodejada per un embolcall fet per les instruccions que acabem de veure.

v

¡Advertiment! Molts dels exemples que apareixen a la documentació del LilyPond ometen les instruccions `\new Staff` i `\new Voice`, deixant que es creïn de forma implícita. Això funciona bé per a exemples senzills, però per a exemples més complicats, especialment quan s’usen instruccions addicionals, la creació implícita dels contextos poden donar lloc a resultats inesperats, fins i tot en ocasions crear pentagrames no desitjats. La forma de crear contextos de forma explícita s’explica a [Secció 3.3 \[Contextos i gravadors\]](#), pàgina 59.

Nota: Quan s’escriuen més d’unes poques línies de música, es recomana crear sempre els pentagrames i les veus de forma explícita.

En tot cas, per ara anem a tornar al primer exemple per examinar l’ordre `\score`, deixant les altres en la seva forma predeterminada.

Un bloc `\score` sempre ha de contenir una expressió musical única, que ha d’aparèixer immediatament després de la instrucció `\score`. Recordeu que una expressió musical pot ser qualsevol cosa, des d’una sola nota fins a una enorme expressió composta com ara

```
{
  \new StaffGroup <<
    ... inseriu aquí la partitura completa d'una òpera
    de Wagner ...
  >>
}
```

A causa que tot es troba dins de `{ ... }`, compta com una expressió musical.

Com hem vist anteriorment, el bloc `\score` pot contenir altres coses, com ara

```
\score {
  { c'4 a b c' }
  \header { }
  \layout { }
  \midi { }
}
```

Observeu que aquestes tres instruccions (`\header`, `\layout` i `\midi`) són especials: a diferència de la resta de les instruccions que comencen amb una barra invertida (`\`), *no* són expressions musicals i no formen part de cap expressió musical. Per tant, es poden posar dins d’un bloc `\score` o a fora d’ell. De fet, aquestes instruccions se situen en general fora del bloc `\score` (per exemple, `\header` se sol col·locar abans de la instrucció `\score`, com mostra l’exemple que apareix al principi de la secció.)

Dues instruccions més que no hem vist són `\layout { }` i `\midi { }`. Si apareixen tal i com es mostren aquí, fan que el LilyPond produeixi una sortida impresa i una sortida MIDI, respectivament. Es descriuen amb tot detall al manual de Referència de la notació, a [Secció “Score layout” in Referència de la notació](#) i a [Secció “Creating MIDI files” in Referència de la notació](#).

Podem escriure diversos blocs `\score`. Cada un d’ells rebrà el mateix tractament que una partitura independent, però es combinaran tots junts a un fitxer de sortida únic. No fa falta cap instrucció `\book`, es crearà un implícitament. No obstant, si voleu fitxers de sortida separats a partir d’un únic fitxer `‘.ly’`, aleshores cal utilitzar l’ordre `\book` per separar les diferents seccions: cada bloc `\book` produeix un fitxer de sortida diferent.

En resum:

Cada bloc `\book` crea un fitxer de sortida diferent (per exemple, un fitxer PDF). Si no hem escrit un de forma explícita, el LilyPond envolta tot el nostre codi d’entrada dins d’un bloc `\book` de forma implícita.

Cada bloc `\score` és un tros de música separat dins d'un bloc `\book`.

Cada bloc `\layout` afecta el bloc `\score` o `\book` dins del qual apareix (és a dir, un bloc `\layout` dins d'un bloc `\score`) afecta solament a aquest bloc `\score`, però un bloc `\layout` fora d'un bloc `\score` (que per això està dins d'un bloc `\book`, ja sigui explícit o implícitament) afecta als blocs `\score` que estan dins d'aquest `\book`.

Per veure més detalls, consulteu [Secció “Multiple scores in a book” in Referència de la notació](#).

Una altra magnífica drecera és la possibilitat de definir variables com es mostra a [Secció 2.4.1 \[Organitzar les peces mitjançant variables\]](#), pàgina 36. Totes les plantilles fan servir el següent:

```
melodia = \relative c' {
  c4 a b c
}

\score {
  \melodia
}
```

Quan el LilyPond examina aquest fitxer, agafa el valor de `melodia` (tot el que hi ha després del signe igual) i l'insereix a tot arreu que veu `\melodia`. No es requereix una cura especial amb el nom (pot ser `melodia`, `global`, `CompasArmadura`, `madretadelpiano` o `fulanet` o qualsevol altre). Recordeu que pot ser gairebé qualsevol nom que se us acudeixi, sempre i quan contingui sols caràcters alfabètics i sigui diferent a qualsevol dels noms d'instrucció del LilyPond. Per veure més detalls, consulteu [Secció 3.4.4 \[Estalviar tecleig mitjançant variables i funcions\]](#), pàgina 84. Les limitacions exactes que afecten els noms de variable es detallen a [Secció “File structure” in Referència de la notació](#).

Vegeu també

Per veure una definició completa del format del codi d'entrada, consulteu [Secció “File structure” in Referència de la notació](#)

3.1.2 La partitura és una (única) expressió musical composta

En la secció anterior, [Secció 3.1.1 \[Introducció a l'estructura dels fitxers del LilyPond\]](#), pàgina 41, hem pogut veure l'organització general dels fitxers d'entrada del LilyPond. Però sembla que ens hem saltat la part més important: com esbrinem què escriure després de `\score`?

No ens hem saltat res de res. El gran misteri és, senzillament, que no hi ha *cap* misteri. La línia següent ho explica tot:

Un bloc `\score` ha de començar amb una expressió musical composta.

Per comprendre el que s'entén per expressió musical i expressió musical composta, potser trobeu útil fer un repàs al tutorial, [Secció 2.2.1 \[Explicació de les expressions musicals\]](#), pàgina 27. En aquesta secció vam veure com elaborar grans expressions musicals a partir de petites peces (començàvem amb notes, després acords, etc.). Ara partirem d'una gran expressió musical i recorrerem el camí invers cap avall. Per simplicitat, farem sols un cantant i un piano. No necessitem un `StaffGroup` (que simplement agrupa un cert nombre de pautes amb un claudàtor a l'esquerra) per a aquest conjunt, i aleshores el retirem. Tot i així, *sí* necessitem pentagrames per a un cantant i un piano.

```
\score {
  <<
    \new Staff = "cantant" <<
    >>
    \new PianoStaff = "piano" <<
    >>
  >>
}
```

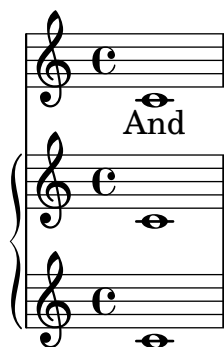


```
\layout { }
}
```

Aquí hem assignat noms als pentagrames: “cantant” i “piano”. Això no és essencial en aquest moment, però és un hàbit que resulta útil cultivar de manera que podem saber d’una ullada per a què és cada pentagrama.

Recordeu que utilitzem `<< ... >>` en comptes de `... }` per presentar la música simultània. Això fa que les parts vocals i del piano apareguin una sobre l’altra a la partitura. La construcció `<< ... >>` no seria necessària per al pentagrama del cantant a l’exemple de dalt si conté solament una expressió musical seqüencial, però es necessitarien els `<< ... >>` en comptes de les claus si la música d’aquest pentagrama anés a contenir dues o més expressions simultànies, per exemple dues veus simultànies, o una veu amb lletra. Tindrem una veu amb lletra, per la qual cosa es requereixen els angles dobles. Després afegirem quelcom de música real; per ara limitem-nos a posar unes quantes notes i text de farciment. Si heu oblidat com afegir la lletra, podeu rellegir la secció `\addlyrics` de [Secció 2.3.1 \[Elaborar cançons senzilles\]](#), pàgina 31.

```
\score {
  <<
    \new Staff = "singer" <<
      \new Voice = "vocal" { c'1 }
      \addlyrics { And }
    >>
    \new PianoStaff = "piano" <<
      \new Staff = "upper" { c'1 }
      \new Staff = "lower" { c'1 }
    >>
  >>
  \layout { }
}
```



Ara tenim molts més detalls. Tenim la pauta del cantant: conté una **Voice** o veu (al LilyPond, aquest terme fa referència a un conjunt de notes, no necessàriament notes vocals – per exemple, un violí generalment toca una veu –) i el text de la cançó. També tenim una pauta de piano: conté un pentagrama superior (mà dreta) i un pentagrama inferior (mà esquerra), tot i que el pentagrama inferior encara no té la clau de Fa.

En aquest moment podríem començar a ficar les notes. Dins dels claudàtors que segueixen a `\new Voice = "vocal"`, podríem començar escrivint

```
\relative c' {
  r4 d8\noBeam g, c4 r
}
```

Però si ho féssim, la secció `\score` es faria força llarga i seria més difícil comprendre el que passa. En comptes d'això utilitzarem identificadors o variables. Recordareu que les vam veure per primer cop a la secció anterior.

Per assegurar-nos que el contingut de la variable `text` s'interpreta com a lletra, el precedim amb `\lyricmode`. Igual que `\addlyrics`, això activa el mode d'entrada de lletra. Sense això, el LilyPond intentaria interpretar el contingut com a notes, cosa que generaria errors (Hi ha d'altres modes, vegeu [Secció "Input modes" in Referència de la notació](#)).

Així doncs, tot escrivint algunes notes, i una clau de Fa per a la mà esquerra, ara tenim un fragment musical de debò:

```
melody = \relative c'' { r4 d8\noBeam g, c4 r }
text    = \lyricmode { And God said, }
upper   = \relative c'' { <g d g,>2~ <g d g,> }
lower   = \relative c { b2 e }

\score {
  <<
    \new Staff = "singer" <<
      \new Voice = "vocal" { \melody }
      \addlyrics { \text }
    >>
    \new PianoStaff = "piano" <<
      \new Staff = "upper" { \upper }
      \new Staff = "lower" {
        \clef "bass"
        \lower
      }
    >>
  >>
  \layout { }
}
```



Quan escriviu una secció `\score` o quan l'esteu llegint, feu-lo a poc a poc i amb cura. Comenceu pel nivell exterior i després treballeu sobre cadascú dels nivells interiors. També és d'una gran ajuda ser molt estricte amb els marges (assegureu-vos que al seu editor de text cada element del mateix nivell comença a la mateixa posició horitzontal).

Vegeu també

Referència de la notació: [Secció "Estructura d'una partitura" in Referència de la notació](#).

3.1.3 Niuat d'expressions musicals

No és essencial declarar tots els pentagrames al començament; es poden crear temporalment en qualsevol moment. Això és d'especial utilitat per crear seccions d'ossia (vegeu [Secció “ossia” in *Glossari musical*](#)). A continuació presentem un exemple senzill que mostra com introduir temporalment un pentagrama nou mentre dura un fragment de tres notes:

```
\new Staff {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
    <<
    { f8 c c }
    \new Staff {
      f8 f c
    }
    >>
    r4 |
  }
}
```



Noteu que la mida de la clau és igual a la que s'imprimeix en un canvi de clau (lleugerament més petita que la clau al principi d'una línia). Això és normal per a qualsevol clau que s'imprimeixi a la meitat d'una línia.

La secció ossia es pot col·locar a sobre del pentagrama de la manera següent:

```
\new Staff = "main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
    <<
    { f8 c c }
    \new Staff \with {
      alignAboveContext = #"main"
    } { f8 f c }
    >>
    r4 |
  }
}
```



Aquest exemple utilitza `\with`, que s'explica en tot detall més endavant. És un mitjà per modificar el comportament predeterminat d'un sol pentagrama. En aquest exemple, diu que el pentagrama nou s'ha de col·locar per sobre del pentagrama anomenat “main” en comptes de la posició predeterminada que seria per sota.

Vegeu també

Els fragments d'ossia s'escriuen sovint sense clau i sense indicació de compàs, i generalment amb una lletra més petita. Per fer això caldrien més ordres que encara no s'han vist. Vegeu [Secció “Mida dels objectes” in *Manual d'aprenentatge*](#) i [Secció “Ossia staves” in *Referència de la notació*](#).

3.1.4 Quant a la impossibilitat de niuar claudàtors i lligadures

A l'escriptura del fitxer d'entrada del LilyPond hem pogut veure alguns tipus de parèntesis, claudàtors i angles de diversos tipus. Obeeixen a diverses regles que el principi poden semblar confuses. Abans d'explicar aquestes regles, fem un repàs a les diverses classes de claudàtors, claus i parèntesis.

Tipus de parèntesis	Funció
<code>{ ... }</code>	Tanca un fragment seqüencial de música
<code>< ... ></code>	Tanca les notes d'un acord
<code><< ... >></code>	Tanca expressions musicals simultànies
<code>(...)</code>	Marca el començament i el final d'una lligadura d'expressió
<code>\(... \)</code>	Marca el començament d'una lligadura de fraseig
<code>[...]</code>	Marca el començament i el final d'un barrat manual

A les anteriors hem d'afegir d'altres construccions que generen línies entre o a través de les notes: les lligadures d'unió (marcades amb un accent corb, `~`), els grups especials que s'escriuen amb `\tuplet x/y { ... }`, i les notes d'adorn, que s'escriuen amb `\grace { ... }`.

Fora del LilyPond, l'ús convencional dels parèntesis i d'altres claudàtors requereix que els diversos tipus es trobin niuats correctament, com a: `<< [{ (...) }] >>`, de manera que els parèntesis que es tanquen han de trobar-se a l'ordre exactament oposat als dels parèntesis que s'obren. Això és un requisit per als tres tipus de parèntesis que es descriuen mitjançant la paraula ‘Tanca’ a la taula anterior: s'han de niuar correctament. Tanmateix, la resta de claus i claudàtors, que estan descrits per la paraula ‘Marca’ a la mateixa taula anterior, **no** han de niuar-se estrictament per cap raó amb cap dels altres parèntesis. De fet, aquests parèntesis no són parèntesis en el sentit que tanquen quelcom: simplement són marcadors que indiquen on comença o finalitza quelcom.

Així doncs, per exemple, una lligadura de fraseig pot començar abans d'una barra inserida manualment, i acabar abans que acabi la barra (una cosa que potser no sigui molt musical, però és possible):

```
g8\ ( a b [ c b \ ) a ] g4
```

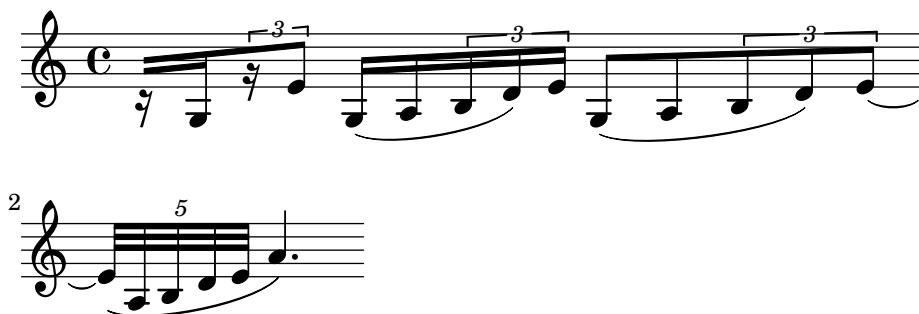


En general, els diversos tipus de claudàtors, i els que es fan servir per grups especials, lligadures d'unió i notes d'adorn, es poden barrejar amb total llibertat. Aquest exemple mostra una barra que s'estén cap a l'interior d'un grup de valoració especial (línia 1), una lligadura d'expressió que es perllonga cap a l'interior d'un grup especial, una lligadura d'unió que travessa dos grups especials, i una lligadura de fraseig que surt de l'interior d'un grup especial (línies 3 i 4).

```

r16[ g \tuplet 3/2 { r16 e'8] }
g,16( a \tuplet 3/2 { b16 d) e }
g,8[( a \tuplet 3/2 { b8 d) e~] } |
\tuplet 5/4 { e32\ ( a, b d e } a4.\)

```



3.2 Les veus contenen música

Igual que els cantants, al LilyPond li calen veus per cantar. En realitat, la música per a qualsevol instrument d'una partitura està sempre continguda dins d'una veu –el concepte del LilyPond més fonamental de tots–.

3.2.1 Escolto veus

De les capes més fondes d'una partitura del LilyPond, les més baixes i més fonamental reben el nom de 'Voice contexts' («contextos de veu») o, abreujadament, 'Voices' («veus»). Les veus s'anomenen a vegades 'layers' («capes») a d'altres programes d'edició de partitures.

De fet, una capa o context de veu és l'única que pot contenir música. Si un context de veu no es declara explícitament, es crea un de forma automàtica, com vam veure al principi d'aquest capítol. Certs instruments com l'oboè sols poden tocar una nota cada cop. La música escrita per a aquests instruments sols requereix una veu. Els instruments que poden tocar més d'una nota a la vegada, com el piano, amb freqüència necessitaran diverses veus per codificar les diverses notes i ritmes concurrents que són capaces de tocar.

Una sola veu pot contenir moltes notes dins d'un acord, per suposat; aleshores, quan, exactament, es necessiten diverses veus? En primer lloc observeu aquest exemple de quatre acords:

```

\key g \major
<d g>4 <d fis> <d a'> <d g>

```



Això es pot expressar utilitzant sols símbols d'acord amb angles simples, < ... >, i per això tan sols es necessita una veu. Però suposeu que el Fa sostingut fos realment una corxera seguida d'un Sol corxera, una nota de pas que condueix al La. Ara tenim dues notes que comencen el mateix moment però tenen diferents duracions: la negra Re, i la corxera Fa sostingut. Com es codifica això? No es poden escriure amb un acord perquè totes les notes d'un acord han de tenir la mateixa duració. I no es poden escriure com dues notes en seqüència perquè han de començar al mateix moment. Aquí és on necessiten dues veus.

Vegem com es fa això dins de la sintaxi d'entrada del LilyPond.

La forma més fàcil d'introduir fragments amb més d'una veu a un sol pentagrama és escriure cada veu com una seqüència (amb { ... }), i combinar-les simultàniament amb angles dobles, << ... >>. Els fragments també s'han de separar mitjançant una doble barra invertida, \,

per situar-los a veus separades. Sense això, les notes anirien a una sola veu, el que normalment produeix errors. Aquesta tècnica s'adapta especialment bé a peces de música que són majorment homofòniques però ocasionalment tenen seccions curtes de polifonia.

Heus ací com dividim els acords anteriors en dues veus i afegim la nota de pas i la lligadura:

```
\key g \major
%      Veu "1"                      Veu "2"
<< { g4 fis8( g) a4 g } \\ { d4 d d d } >>
```



Observe com les pliques de la segona veu ara es dirigeixen cap avall.

A continuació vegem un altre exemple senzill:

```
\key d \minor
%      Veu "1"                      Veu "2"
<< { r4 g g4. a8 } \\ { d,2 d4 g } >> |
<< { bes4 bes c bes } \\ { g4 g g8( a) g4 } >> |
<< { a2. r4 } \\ { fis2. s4 } >> |
```



No és necessari usar una construcció << \\ >> diferent per a cada compàs. Per a música que tingui unes poques notes a cada compàs, aquesta disposició podria facilitar la llegibilitat del codi, però si hi ha moltes notes a cada compàs podria ser millor dividir-lo en dues veus separades de la següent manera:

```
\key d \minor
<< {
  % Veu "1"
  r4 g g4. a8 |
  bes4 bes c bes |
  a2. r4 |
} \\ {
  % Veu "2"
  d,2 d4 g |
  g4 g g8( a) g4 |
  fis2. s4 |
} >>
```



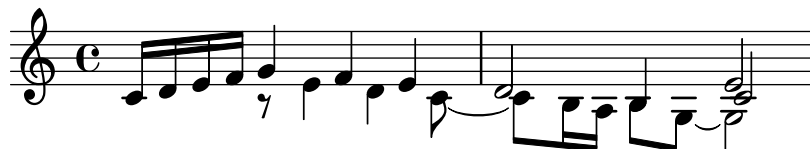
Aquest exemple té sols dues veus, però la mateixa construcció es pot usar per codificar tres o més veus mitjançant l'addició de més separadors de barra invertida.

Els contextos de veu porten els noms de "1", "2", etc. Els primers contextos estableixen les veus *externes*, la veu aguda del context "1" i la veu greu del context "2". Les veus interiors van als contextos "3" i "4". A cada un d'aquests contextos, la direcció vertical de les lligadures, pliques, matisos dinàmics, etc., s'ajusta de forma correcta.

```

\new Staff \relative c' {
  % Veu principal
  c16 d e f
  %      Veu "1"          Veu "2"          Veu "3"
  << { g4 f e } \\\ { r8 e4 d c8~ } >> |
  << { d2 e } \\\ { c8 b16 a b8 g~ 2 } \\\ { s4 b c2 } >> |
}

```

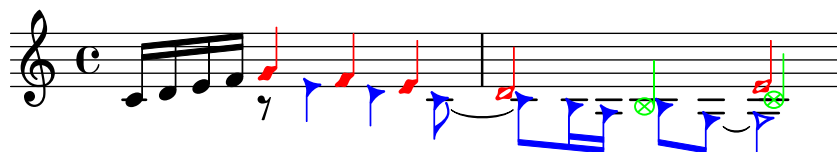


Totes aquestes veus estan separades de la veu principal que conté les notes just per fora de la construcció << ... >>. Anomenarem això la *construcció simultània*. Les lligadures (de prolongació i d'expressió) solament poden connectar notes que estiguin dins de la mateixa veu, en conseqüència les lligadures no poden entrar o sortir d'una construcció simultània. A la inversa, les veus paral·leles de construccions simultànies diferents sobre el mateix pentagrama són la mateixa veu. Hi ha altres propietats relatives a les veus que també impliquen construccions simultànies. A continuació veiem el mateix exemple, amb colors i caps diferents per a cada veu. Observeu que els canvis a una veu no afecten a d'altres veus, però persisteixen més tard dins de la mateixa veu. Observeu també que les notes lligades es poden dividir entre les mateixes veus de dues construccions, com s'indica aquí a la veu de triangles blaus.

```

\new Staff \relative c' {
  % Veu principal
  c16 d e f
  << % Bar 1
  {
    \voiceOneStyle
    g4 f e
  }
  \\\
  {
    \voiceTwoStyle
    r8 e4 d c8~
  }
  >> |
  << % Bar 2
    % Continua la Veu 1
    { d2 e }
  \\\
    % Continua la Veu 2
    { c8 b16 a b8 g~ 2 }
  \\\
  {
    \voiceThreeStyle
    s4 b c2
  }
  >> |
}

```



Les instruccions `\voiceXXXStyle` estan pensades principalment per usar-les en documents educatius com el que presentem aquí. Modifiquen el color del cap, la plica i les barres, i l'estil del cap, de manera que les veus es poden distingir fàcilment. La veu u està establerta a rombes vermells, la veu dos a triangles blaus, la veu tres a cercles verds amb aspes, i la veu quatre (que no es fa servir aquí) a aspes color magenta. `\voiceNeutralStyle` (que tampoc no es fa servir aquí) retorna tot a l'estil predeterminat. Veurem més endavant com l'usuari pot crear instruccions com aquestes. Vegeu [Secció “Visibilitat i color dels objectes”](#) in *Manual d'aprenentatge* i [Secció “Ús de variables per als ajustos de disposició”](#) in *Manual d'aprenentatge*.

La polifonia no canvia la relació de les notes dins d'un bloc `\relative`. L'alçada de cada nota encara es calcula amb relació a la nota que la precedeix immediatament, o a la primera nota de l'acord precedent. Així, en

```
\relative c' { notaA << < notaB notaC > \\\ notaD >> notaE }
```

`notaB` és relativa a `notaA`

`notaC` és relativa a `notaB`, no a `notaA`;

`notaD` és relativa a `notaB`, no a `notaA` ni a `notaC`;

`notaE` és relativa a `notaD`, no a `notaA`.

Una forma alternativa, que podria ser més clara si les notes a les veus estan molt separades, és col·locar una instrucció `\relative` al principi de cada veu:

```
\relative c' { notaA ... }
<<
  \relative c'' { < notaB notaC > ... }
\\
  \relative g' { notaD ... }
>>
\relative c' { notaE ... }
```

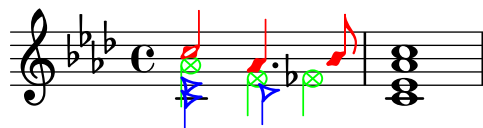
Finalment, analitzem les veus en una peça de música més complexa. Heus aquí les notes dels dos primers compassos del segon dels Dos Nocturns de Chopin, Op 32. Aquest exemple s'utilitzarà en fases posteriors dins del present capítol i el següent, per il·lustrar diverses tècniques per produir notació, i per tant us demanem que ignoreu per ara qualsevol cosa al codi subjacent que li sembli misteriós i tan sols es concentri a la música i les veus (totes les complicacions s'explicaran a seccions posteriors).



Amb freqüència, la direcció de les pliques s'utilitza per indicar la continuïtat de dues línies melòdiques simultànies. En aquest cas, totes les pliques de les notes agudes es dirigeixen cap amunt i les de la notes greus cap avall. Aquesta és la primera indicació que es requereix més d'una veu.

Però la necessitat real de diverses veus apareix quan hi ha notes que comencen al mateix temps però tenen duracions diferents. Observeu les notes que comencen a la tercera part del primer compàs. El La bemoll és una negra amb puntet, el Fa és una negra i el Re bemoll és una blanca. Aquestes notes no es poden escriure com un acord perquè totes les notes d'un acord han de tenir la mateixa duració. Tampoc es poden escriure com notes seqüencials, ja que han de començar al mateix temps. Aquesta secció del compàs requereix tres veus, i la pràctica comuna seria escriure tot el compàs com tres veus com es mostra a sota, on hem usat diferents caps i

colors per a la tres veus. Un cop més, el codi que hi ha a aquest exemple s'explicarà més tard, així que ignoreu tot el que no entengueu.



Intentarem codificar aquesta música partint de zero. Com veurem, això s'enfronta a certes dificultats. Començarem tal com hem après, usant la construcció `<< \ \ >>` per introduir la música del primer compàs a tres veus:

```
\new Staff \relative c'' {
  \key aes \major
  <<
    { c2 aes4. bes8 } \ \ { <ees, c>2 des } \ \ { aes'2 f4 fes }
  >> |
  <c ees aes c>1 |
}
```



Les direccions de les pliques s'assignen automàticament de forma que les veus de numeració imparell reben les pliques cap amunt i les de numeració parell cap avall. Les pliques de les veus 1 i 2 són correctes, però les pliques de la veu 3 haurien d'anar cap avall en aquest fragment en particular. Podem corregir això simplement oblidant-nos de la veu tres i situant la música a la veu quatre. Això es fa escrivint un altre parell de barres invertides (`\ \`).

```
\new Staff \relative c'' {
  \key aes \major
  << % Veu u
    { c2 aes4. bes8 }
  \ \ % Veu dos
    { <ees, c>2 des }
  \ \ % Omet Veu tres
  \ \ % Veu quatre
    { aes'2 f4 fes }
  >> |
  <c ees aes c>1 |
}
```



Veiem que això arregla la direcció de la plica, però la col·locació horitzontal de les notes no és la desitjada. El LilyPond desplaça les notes interiors quan elles o les seves pliques d'una altra manera col·lisionarien amb les veus exteriors, però això no és el més adequat per a música de piano. En altres situacions, els desplaçaments que el LilyPond aplica poden no eliminar les col·lisions. El LilyPond aporta diverses formes d'ajustar la col·locació horitzontal de les notes. Encara no estem preparats per veure com es corregeix això, per la qual cosa deixarem el problema

per a una secció posterior (vegeu la propietat `force-hshift` a Secció “Arreglar notació amb superposicions” in *Manual d’aprenentatge*).

Nota: No es poden crear lletres ni objectes d’extensió (com ara lligadures, reguladors, etc.) ‘entre’ veus diferents.

Vegeu també

Referència de la notació: Secció “Multiple voices” in *Referència de la notació*.

3.2.2 Veus explícites

Els contextos de veu també es poden crear manualment dins d’un bloc `<< >>` per crear música polifònica, utilitzant `\voiceOne` ... `\voiceFour` per indicar les direccions requerides de pliques, lligadures, etc. A partitures més llargues, aquest mètode és més clar perquè permet que les veus estiguin separades i rebin noms més descriptius.

Concretament, la construcció `<< \ \ >>` que usem a la secció prèvia:

```
\new Staff {
  \relative c' {
    << { e4 f g a } \ \ { c,4 d e f } >>
  }
}
```

equivale a

```
\new Staff <<
  \new Voice = "1" { \voiceOne \relative c' { e4 f g a } }
  \new Voice = "2" { \voiceTwo \relative c' { c4 d e f } }
>>
```

Els dos exemples anteriors produeixen:



Les instruccions `\voiceXXX` estableixen la direcció de les pliques, lligadures d’expressió, lligadures de prolongació, articulacions, anotacions de text, puntets i digitacions. `\voiceOne` i `\voiceThree` fan que aquests objectes apuntin cap amunt, mentre que `\voiceTwo` i `\voiceFour` els fan apuntar cap avall. Aquestes instruccions també produeixen un desplaçament horitzontal per a cada veu quan és necessari per evitar xocs entre els caps. La instrucció `\oneVoice` retorna els ajustos de nou als valors normals per a una sola veu.

Vegem a alguns exemples senzills exactament quin efecte tenen `\oneVoice`, `\voiceOne` i `\voiceTwo` sobre l’escriptura, les lligadures d’unió i d’expressió i les indicacions de dinàmica:

```
\relative c' {
  % Comportament predeterminat o comportament després de \oneVoice
  c4 d8~ 8 e4( f | g4 a) b-> c |
}
```

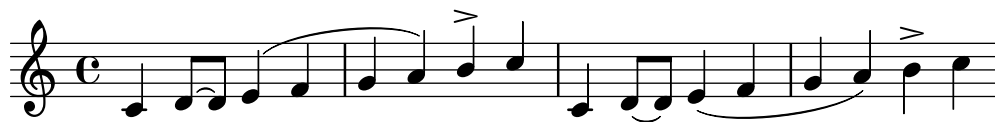


```
\relative c' {
```

```

\voiceOne
c4 d8~ 8 e4( f | g4 a) b-> c |
\oneVoice
c,4 d8~ 8 e4( f | g4 a) b-> c |
}

```



```

\relative c' {
  \voiceTwo
  c4 d8~ 8 e4( f | g4 a) b-> c |
  \oneVoice
  c,4 d8~ 8 e4( f | g4 a) b-> c |
}

```



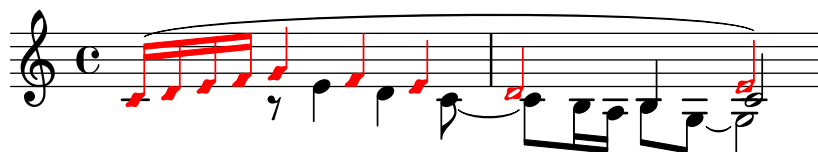
A continuació veurem tres formes diferents de compondre la notació del mateix passatge polifònic, cada una de les quals té els seus avantatges segons la circumstància, utilitzant l'exemple de la secció anterior.

Una expressió que apareix directament dins de << >> pertany a la veu principal (però, observeu, **no** dins d'una construcció << \ \ >>). Això és útil quan apareixen veus noves mentre la veu principal està sonant. A continuació podem veure una realització més correcta de l'exemple de la secció anterior. Les notes vermelles en forma de rombe mostren que la melodia principal està ara dins d'un context d'una sola veu, fent que es pugui traçar una lligadura per sobre d'elles.

```

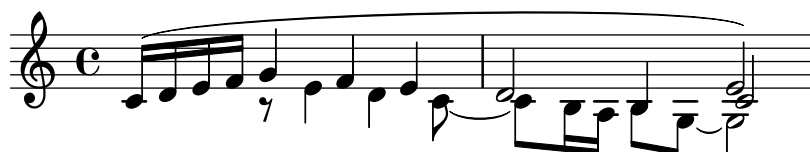
\new Staff \relative c' {
  \voiceOneStyle
  % Aquesta secció és homofònica
  c16^( d e f
  % Comença una secció simultània de tres veus
  <<
    % Continua la veu principal en paral.lel
    { g4 f e | d2 e) | }
    % Inicia la segona veu
    \new Voice {
      % Estableix les pliques, etc., cap avall
      \voiceTwo
      r8 e4 d c8~ | 8 b16 a b8 g~ 2 |
    }
    % Inicia la tercera veu
    \new Voice {
      % Set stems, etc, up
      \voiceThree
      s2. | s4 b c2 |
    }
  >>
}

```



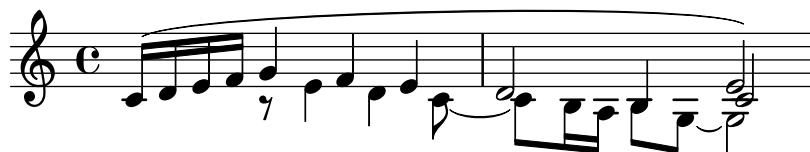
Les construccions polifòniques niuades més profundament són possibles, i si hi ha una veu que apareix sols breument podria haver-hi una forma més natural d'escriure la música.

```
\new Staff \relative c' {
  c16^( d e f
  <<
    { g4 f e | d2 e) | }
  \new Voice {
    \voiceTwo
    r8 e4 d c8~ |
    <<
      { c8 b16 a b8 g~ 2 | }
    \new Voice {
      \voiceThree
      s4 b c2 |
    }
  }
  >>
}
>>
}
```



Aquest mètode de niuar veus noves breument és útil quan sols hi ha seccions polifòniques petites, però quan tot el pentagrama és molt polifònic podria ser més clar usar sempre diverses veus, usant notes espaiadores per passar per sobre de les seccions on una veu està en silenci, com per exemple:

```
\new Staff \relative c' <<
  % Inicia la primera veu
  \new Voice {
    \voiceOne
    c16^( d e f g4 f e | d2 e) |
  }
  % Inicia la segona veu
  \new Voice {
    % Estableix les pliques, etc., cap avall
    \voiceTwo
    s4 r8 e4 d c8~ | 8 b16 a b8 g~ 2 |
  }
  % Inicia la tercera veu
  \new Voice {
    % Estableix les pliques, etc., cap amunt
    \voiceThree
    s1 | s4 b c2 |
  }
  >>
```



Columnes de notes

Les notes properes a un acord, o les notes que es produeixen al mateix temps en diferents veus, es disposen en dos (i ocasionalment més) columnes per evitar el solapament dels caps. Reben el nom de columnes de notes. Hi ha columnes diferents per a cada veu, i el desplaçament especificat en curs depenent de la veu s'aplica a la columna de la nota si en cas contrari es produís una col·lisió. Això es pot veure a l'exemple anterior. Al compàs 2 el Do a la veu dos està desplaçat a la dreta respecte del Re de la veu u, i a l'últim acord el Do de la veu tres també està desplaçat a la dreta respecte de les altres notes.

Les instruccions `\shiftOn`, `\shiftOnn`, `\shiftOnnn` i `\shiftOff` especifiquen el grau que s'han de desplaçar les notes i acords de la veu si en cas contrari es produís una col·lisió. De forma predeterminada, les veus exteriors (normalment les veus u i dos) tenen com a especificació `\shiftOff`, mentre que les veus interiors (tres i quatre) tenen especificat `\shiftOn`. Quan s'aplica un desplaçament, les veus u i tres es desplacen cap a la dreta i les veus dos i quatre es desplacen cap a l'esquerra.

`\shiftOnn` i `\shiftOnnn` defineixen nivells addicionals de desplaçament que es poden especificar temporalment per resoldre col·lisions en situacions complexes (vegeu [Secció "Exemples reals de música"](#) in *Manual d'aprenentatge*).

Una columna de notes pot contenir sols una nota (o acord) d'una veu amb les pliques cap amunt i una nota (o acord) d'una veu amb les pliques cap avall. Si les notes de dues veus que tenen les pliques a la mateixa direcció se situen en la mateixa posició i les dues veus no tenen cap desplaçament o porten especificat el mateix desplaçament, es produirà el missatge d'error "Xoquen massa columnes de notes".

Vegeu també

Manual d'aprenentatge: [Secció "Moure objectes"](#) in *Manual d'aprenentatge*.

Referència de la notació: [Secció "Multiple voices"](#) in *Referència de la notació*.

3.2.3 Veus i música vocal

La música vocal presenta una dificultat especial: hem de combinar dues expressions, és a dir, les notes i la lletra.

Ja heu vist la instrucció `\addlyrics{}`, que funciona bé per a partitures senzilles. Tot i així, aquesta tècnica és una mica limitada. Per a música de complexitat més gran, hem d'introduir la lletra en un context `Lyrics` utilitzant `\new Lyrics` i enllaçar explícitament la lletra i les notes mitjançant `\lyricsto{}`, usant el nom assignat a la veu.

```
<<
\new Voice = "una" {
  \relative c'' {
    \autoBeamOff
    \time 2/4
    c4 b8. a16 | g4. f8 | e4 d | c2 |
  }
}
\new Lyrics \lyricsto "una" {
  No more let | sins and | sor -- rows | grow. |
}
>>
```



Observeu que la lletra s'ha d'enllaçar a un context de **Voice**, *no* a un context de **Staff**. Aquest és un cas on és necessari crear contextos de **Staff** i de **Voice** explícitament.

El barrat automàtic que el LilyPond usa de forma predeterminada funciona bé per a la música instrumental, però no tan bé per a la música sense lletra, on o bé el barrat no es necessita en absolut, o bé s'utilitza per indicar els melismes de la lletra. A l'exemple anterior hem utilitzat la instrucció `\autoBeamOff` per desactivar el barrat automàtic.

Ara reutilitzarem l'exemple anterior de «Judes Macabeu» per a il·lustrar aquesta tècnica més flexible. Primer la reescriurem per que faci servir variables, de manera que la música i la lletra es puguin separar de l'estructura de pentagrames. També introduïrem una clau de grup de **ChoirStaff**. La lletra pròpiament dita s'ha de introduir amb `\lyricmode` perquè tenir seguretat que s'interpreti com a lletra i no com a música.

```
global = { \key f \major \time 6/8 \partial 8 }

SopOneMusic = \relative c'' {
  c8 | c8([ bes]) a a([ g]) f | f'4. b, | c4.~ 4
}
SopOneLyrics = \lyricmode {
  Let | flee -- cy flocks the | hills a -- dorn, --
}
SopTwoMusic = \relative c' {
  r8 | r4. r4 c8 | a'8([ g]) f f([ e]) d | e8([ d]) c bes'
}
SopTwoLyrics = \lyricmode {
  Let | flee -- cy flocks the | hills a -- dorn,
}

\score {
  \new ChoirStaff <<
    \new Staff <<
      \new Voice = "SopOne" {
        \global
        \SopOneMusic
      }
      \new Lyrics \lyricsto "SopOne" {
        \SopOneLyrics
      }
    >>
  \new Staff <<
    \new Voice = "SopTwo" {
      \global
      \SopTwoMusic
    }
    \new Lyrics \lyricsto "SopTwo" {
      \SopTwoLyrics
    }
  >>
}
>>
```



Aquesta és l'estructura bàsica de totes les partitures vocals. Es poden afegir més pentagrames segons es necessiti, es poden afegir més veus als pentagrames i més estrofes a la lletra, i les variables que contenen la música es poden col·locar fàcilment en fitxers separats quan es facin massa llargs.

A continuació podem veure un exemple final de la primera línia d'un himne amb quatre estrofes, per a cor SATB. En aquest cas la lletra de les quatre parts és la mateixa. Observeu com utilitzem variables per a separar la notació musical de l'estructura de pentagrames. Observeu també com s'utilitza una variable, per a la qual hem escollit el nom 'TimeKey' («compàs i tonalitat»), per a que contingui diverses instruccions que s'usaran dins dels dos pentagrames. A d'altres exemples se li sol donar el nom de 'global'.

```
keyTime = { \key c \major \time 4/4 \partial 4 }

SopMusic  = \relative c' { c4 | e4. e8 g4 g | a4 a g }
AltoMusic  = \relative c' { c4 | c4. c8 e4 e | f4 f e }
TenorMusic = \relative c { e4 | g4. g8 c4. b8 | a8 b c d e4 }
BassMusic  = \relative c { c4 | c4. c8 c4 c | f8 g a b c4 }

VerseOne =
  \lyricmode { E -- | ter -- nal fa -- ther, | strong to save, }
VerseTwo  =
  \lyricmode { O | Christ, whose voice the | wa -- ters heard, }
VerseThree =
  \lyricmode { O | Ho -- ly Spi -- rit, | who didst brood }
VerseFour  =
  \lyricmode { O | Tri -- ni -- ty of | love and pow'r }

\score {
  \new ChoirStaff <<
    \new Staff <<
      \clef "treble"
      \new Voice = "Sop" { \voiceOne \keyTime \SopMusic }
      \new Voice = "Alto" { \voiceTwo \AltoMusic }
      \new Lyrics \lyricsto "Sop" { \VerseOne }
      \new Lyrics \lyricsto "Sop" { \VerseTwo }
      \new Lyrics \lyricsto "Sop" { \VerseThree }
      \new Lyrics \lyricsto "Sop" { \VerseFour }
    >>
    \new Staff <<
      \clef "bass"
      \new Voice = "Tenor" { \voiceOne \keyTime \TenorMusic }
      \new Voice = "Bass" { \voiceTwo \BassMusic }
    >>
  >>
}
```



Vegeu també

Referència de la notació: *Secció “Vocal music” in Referència de la notació.*

3.3 Contextos i gravadors

Els contextos i els gravadors s’han mencionat de manera informal a seccions anteriors; ara tan sols veurem aquests conceptes amb més detall, ja que són importants a l’ajust fi de la sortida del LilyPond.

3.3.1 Explicació dels contextos

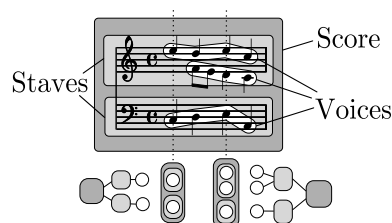
Quan s’imprimeix la música, s’han de afegir a la sortida una gran quantitat d’elements de notació que no apareixen explícitament al fitxer d’entrada. Per exemple, compareu l’entrada i la sortida del següent exemple:

```
cis4 cis2. | a4 a2. |
```



L’entrada és força greu, però a la sortida s’han afegit línies divisòries, les alteracions, la clau i l’armadura de la tonalitat. Quan el LilyPond *interpreta* l’entrada, la informació musical s’analitza d’esquerra a dreta de la mateixa manera que un intèrpret llegeix la partitura. Mentre es llegeix el codi d’entrada, el programa recorda on estan els límits dels compassos, i quines alçades requereixen alteracions accidentals explícites. Aquesta informació s’ha de conservar a diversos nivells. Per exemple, una alteració accidental afecta solament a un pentagrama, mentre que una línia divisòria ha d’estar sincronitzada al llarg de tot els sistema.

Dins del LilyPond, aquestes regles i petites porcions d’informació s’agrupen en *Contexts*. Ja hem vist el context de veu, **Voice**. Altres exemples de contextos són **Staff** (Pauta o pentagrama) i **Score** (Partitura). Els contextos són jeràrquics, de forma que reflecteixen la naturalesa jeràrquica d’una partitura musical. Per exemple: un context de **Staff** pot contenir molts contextos de **Voice**, i un context de **Score** pot contenir molts contextos de **Staff**.



Cada context assumeix la responsabilitat d’imposar algunes regles de notació, creant certs objectes de notació i mantenint les propietats associades. Per exemple, el context **Voice** pot

introduir una alteració accidental i llavors el context **Staff** manté la regla de mostrar o suprimir l'alteració per a la resta del compàs.

Un altre exemple el constitueix el fet que la sincronització de les línies divisòries es gestiona dins del context de la partitura, **Score**, de forma determinada. Nogensmenys, a algunes músiques és possible que vulguem que les línies divisòries estiguin sincronitzades (pensem en una partitura polimètrica en compassos de 4/4 i de 3/4). En aquests casos hem de modificar els ajustos per omissió dels contextos **Score** i **Staff**.

Per a partitures molt senzilles, els contextos es creen implícitament i no hem de preocupar-nos per ells. Per a peces més grans, com per exemple qualsevol que tingui més d'un pentagrama, els contextos s'han de crear explícitament per assegurar-nos que no tindrem la quantitat exacta de pentagrames que necessitem, i que estan a l'ordre correcte. Per escriure peces amb notació especialitzada, és freqüent la modificació de contextos existents o fins i tot definir d'uns completament nous.

A més dels contextos **Score**, **Staff** i **Voice**, hi ha contextos que se situen entre els nivells de partitura i de pentagrama per controlar els grups de pentagrames, com els contextos alternatius de pentagrama i de veu, i contextos per a la lletra, la percussió, diagrames de trasts, baix xifrat, etc.

Els noms de tots els tipus de contextos es componen d'una o més paraules que comencen amb majúscula i que estan unides unes a les altres sense guió ni barra baixa, per exemple: **PartituraDeTranscripcióGregoriana**

Vegeu també

Referència de la notació: **Secció “Explicació dels contextos”** in *Referència de la notació*.

3.3.2 Creació de contextos

A un fitxer d'entrada, el bloc de partitura, que es presenta precedit per la instrucció `\score`, conté una sola expressió musical i una definició de sortida associada (o bé un bloc `\layout` o bé un bloc `\midi`). El context **Score** se sol deixar que es creï automàticament quan comença la interpretació d'aquesta expressió musical.

Per a partitures que solament tenen una veu i un pentagrama, podem també deixar que els contextos **Voice** i **Staff** se creïn automàticament, però per a partitures més complexes és necessari crear-los a mà. La instrucció més simple que fa això és `\new`. S'anteposa a una expressió musical, per exemple

```
\new tipus expressió_musical
```

on *tipus* és el nom d'un context (com **Staff** o **Voice**). Aquesta instrucció crea un context nou, i comença a interpretar la *expressió_musical* que està dins d'aquest context.

Nota: No s'ha d'usar `\new Score` perquè el context **Score** essencial del nivell superior ja es crea automàticament al interpretar-se l'expressió musical que està dins del bloc `\score`. Els valors predeterminats de propietats de context vàlids per a tota la partitura es poden canviar dins del bloc `\layout`. Vegeu **Secció 3.3.4 [Modificar les propietats dels contextos]**, pàgina 63.

En les seccions anteriors heu pogut veure molts exemples pràctics que creaven nous contextos de **Staff** i de **Voice**, però per a recordar-vos com s'usen aquestes instruccions en la pràctica, heus aquí un exemple anotat de música real:

```
\score { % inici de l'expressió única de música composta
  << % inici d'una secció pentagrames simultanis
    \time 2/4
```

```

\new Staff { % crea el pentagrama de la mà dreta
  \clef "treble"
  \key g \minor
  \new Voice { % crea la veu per a les notes de la mà dreta
    \relative c'' { % inici de les notes de la mà dreta
      d4 ees16 c8. |
      d4 ees16 c8. |
    } % fi de les notes de la mà dreta
  } % fi de la veu de la mà dreta
} % fi del pentagrama de la mà dreta
\new Staff << % crea el pentagrama de la mà esquerra
  \clef "bass"
  \key g \minor
  \new Voice { % crea la veu de la mà esquerra
    \voiceOne
    \relative g { % inicia les notes de la veu u de la mà esquerra
      g8 <bes d> ees, <g c> |
      g8 <bes d> ees, <g c> |
    } % fi de les notes de la veu u de la mà esquerra
  } % fi de la veu u de la mà esquerra
  \new Voice { % crea la veu dos de la mà esquerra
    \voiceTwo
    \relative g { % inici de les notes de la veu dos de la mà esquerra
      g4 ees |
      g4 ees |
    } % fi de les notes de la veu dos de la mà esquerra
  } % fi de la veu dos de la mà esquerra
>> % fi del pentagrama de la mà esquerra
>> % fi de la secció de pentagrames simultanis
} % fi de l'expressió única de música composta

```



(Observeu com totes les instruccions que obren un bloc amb un claudàtor corb, {, o amb angles dobles, <<, estan sagnades (tenen un marge addicional) amb dos espais addicionals, i el claudàtor de tancament corresponent té un marge exactament igual. Tot i no ser necessari, observar aquesta pràctica reduirà considerablement el nombre d'errors de 'parèntesis descompensats', i es recomana especialment. Permet apreciar d'una sola ullada l'estructura de la música, i qualsevol parèntesis descompensat apareixerà molt clarament. Observeu també com el pentagrama de la mà esquerra es crea usant dobles angles perquè requereix dues veus, mentre que el pentagrama de la mà dreta es crea amb una expressió musical única tancada entre claudàtors perquè sols requereix una veu.)

La instrucció `\new` també pot atorgar un nom identificatiu al context per distingir-lo d'altres contextos del mateix tipus:

```
\new tipus = identificador expressió_musical
```

Observeu la distinció entre el nom del tipus de context, **Staff**, **Voice**, etc., i el nom identificatiu d'una instància en particular d'aquest tipus, que pot ser qualsevol seqüència de lletres inventada per l'usuari. En el nom identificatiu també es poden utilitzar dígit i espais, però en aquest cas aquest nom ha d'anar entre cometes, per exemple `\new Staff = "ElMeuPentagrama 1" expressió_musical`. El nom identificatiu s'utilitza per referir-nos més tard a aquesta instància en particular d'un context. Hem vist això a la secció sobre la lletra, a [Secció 3.2.3 \[Veus i música vocal\]](#), pàgina 56.

Vegeu també

Referència de la notació: [Secció “Creating contexts” in Referència de la notació](#).

3.3.3 Explicació dels gravadors

Totes i cadascuna de les marques de la sortida impresa d'una partitura feta amb el LilyPond està produïda per un **Engraver** (gravador). Així, tenim un gravador per imprimir pentagrames, un altre per imprimir els caps de les notes, un altre per a les pliques, un altre per a les barres, i molts més. En total hi ha més de 120 gravadors! Afortunadament, per a la major part de les partitures no és necessari conèixer més que alguns, per a partitures senzilles no hem de saber res de cap d'ells.

Els gravadores resideixen i operen dins de Contextos. Els gravadors com ara el gravador de la indicació de metrònom, **Metronome_mark_engraver**, l'acció i resultat del qual s'apliquen a la partitura com un tot, operen en el context més alt: el context de partitura **Score**.

El gravador de la clau **Clef_engraver** i el de l'armadura **Key_engraver** es troben probablement a tots els contextos de pentagrama (**Staff**), ja que els diferents pentagrames podrien requerir diferents claus i armadures.

El gravador dels caps de nota **Note_heads_engraver** i el de les pliques **Stem_engraver** viuen en cada u dels contextos de veu **Voice**, el context de nivell més baix de tots.

Cada gravador processa els objectes particulars associats amb la seva funció, i manté les propietats que estan relacionades amb aquesta funció. Aquestes propietats, com les que estan associades amb els contextos, es poden modificar per canviar el funcionament del gravador o l'aspecte d'aquests elements de la partitura impresa.

Tots els gravadors tenen noms compostos de diverses paraules que descriuen la seva funció. Sols està en majúscules la inicial de la primera paraula, i la resta se li uneix mitjançant guionets baixos. D'aquesta manera el gravador **Staff_symbol_engraver** és responsable de la creació de les línies del pentagrama, i el **Clef_engraver** determina i estableix l'alçada o el punt de referència sobre el pentagrama dibuixant un símbol de clau.

A continuació presentem alguns dels gravadors més comuns, junt amb la seva funció. Podreu comprovar que és fàcil endevinar la funció a partir del nom (en anglès), i a l'inrevés.

Gravador	Funció
Accidental_engraver	Fa les alteracions accidentals, de precaució i de suggeriment.
Beam_engraver	Grava les barres
Clef_engraver	Grava les claus
Completion_heads_engraver	Divideix les notes que travessen una línia divisòria
Dynamic_engraver	Crea reguladors i indicacions dinàmiques textuals
Forbid_line_break_engraver	Evita els salts de línia si queda algun element musical actiu
Key_engraver	Crea l'armadura de la tonalitat
Metronome_mark_engraver	Grava la indicació de metrònom

Note_heads_engraver	Grava el cap de les notes
Rest_engraver	Grava els silencis
Staff_symbol_engraver	Grava les cinc línies (de forma predeterminada) del pentagrama
Stem_engraver	Crea les pliques i els trèmols d'una sola plica
Time_signature_engraver	Crea les indicacions de compàs

Més endavant veurem es pot canviar com la sortida del LilyPond mitjançant la modificació del funcionament dels Gravadors.

Vegeu també

Referència de funcionament intern: [Secció “Engravers and Performers” in Referència de funcionament intern.](#)

3.3.4 Modificar les propietats dels contextos

Els contextos es responsabilitzen de mantenir els valors d'un cert nombre de *properties* de context. Moltes d'elles es poden canviar per influir en la interpretació del codi d'entrada i canviar així l'aparença de la sortida impresa. Es modifiquen mitjançant la instrucció `\set`. Aquesta instrucció pren la forma següent:

```
\set NomDelContext.nomDeLaPropietat = #valor
```

On el *NomDelContext* és normalment **Score**, **Staff** o **Voice**. Es pot ometre, i en aquest cas se suposa que és el context en curs (normalment **Voice**).

Els noms de les propietats de context consisteixen en paraules unides sense cap guió o barra fixa, i on totes les paraules excepte la primera comencen en majúscula. A continuació podem veure alguns exemples de noms de propietats utilitzades amb freqüència. Hi ha moltes més que les que es mostren aquí.

nomDeLaPropietat	Tipus	Funció	Valor d'exemple
extraNatural	Booleà	Si és vertader, posa becaires addicionals abans de les alteracions	#t , #f
currentBarNumber	Enter	Ajustar el número del compàs actual	50
doubleSlurs	Booleà	Si és vertader, imprimir lligadures d'expressió per sobre i per sota de les notes	#t , #f
instrumentName	Text	Establir el nom del pentagrama, situat a l'esquerra	"Cello I"
fontSize	Real	Augmentar o disminuir la mida de la font tipogràfica	2.4
stanza	Text	Establir el text que s'imprimeix abans del començament d'una estrofa	"2"

on un valor Booleà es vertader (**#t**, True) o fals (**#f**, False), un Enter és un nombre enter positiu, un nombre real és un nombre decimal positiu o negatiu, i el text es tanca entre cometes dobles. Observeu l'aparició de signes de coixinet (**#**), en dos llocs diferents: com a part del valor Booleà abans de la **t** o la **f**, i abans del *valor* dins de la instrucció `\set`. Així doncs, quan s'està escrivint un valor Booleà, s'han d'escriure dos signes de coixinet, per exemple: **##t**.

Abans de poder establir qualsevol d'aquestes propietats, hem de saber en quin context operen. A vegades és quelcom obvi, però en ocasions pot ser quelcom complicat. Si especifiquem un context equivocat, no es produeix cap missatge d'error, però el funcionament esperat no tindrà lloc. Per exemple, la propietat **instrumentName** (nom de l'instrument) viu clarament dins del

context de **Staff**, ja que és el pentagrama el que ha de ser anomenat. En aquest exemple, el primer pentagrama resulta etiquetat, però no el segon, perquè hem omès el nom del context.

```
<<
\new Staff \relative c'' {
  \set Staff.instrumentName = #"Soprano"
  c2 c
}
\new Staff \relative c' {
  \set instrumentName = #"Alto" % Incorrecte!
  d2 d
}
>>
```



Recordeu que el nom del context predeterminat és **Voice**, així que la segona instrucció `\set` estableix la propietat `instrumentName` del context **Voice** a “Alto”, però com el LilyPond no busca aquesta propietat al context **Voice**, no es realitza cap acció. Això no és un error, i no es registra cap missatge al fitxer Log del registre d’errors.

De forma semblant, si el nom de la propietat s’escriu amb alguna falta, no es produeix cap missatge d’error, i clarament l’acció esperada no pot tenir lloc. De fet, es pot establir qualsevol ‘property’ (fictícia) usant qualsevol nom que volem en qualsevol context que existeixi, mitjançant l’ús de la instrucció `\set`. Però si el nom no és conegut per al LilyPond, no produirà cap acció. Alguns editors de text que donen suport als fitxers d’entrada del LilyPond de manera especial, documenten els noms de propietats amb vinyetes quan passem sobre ells el punter del ratolí, com ara JEdit amb l’extensió LilyPondTool, o destaquen els noms de propietats desconegudes de manera diferent, com ara ConTEXT. Si no s’utilitza un editor amb aquesta possibilitats, es recomana comprovar la correcció del nom de la propietat al manual de Referència de funcionament intern: vegeu *Secció “Tunable context properties” in Referència de funcionament intern* o *Secció “Contexts” in Referència de funcionament intern*.

La propietat `instrumentName` tindrà efecte solament si s’estableix dins del context **Staff**, però algunes propietats es poden establir a més d’un context. Per exemple, la propietat `extraNatural` està establerta de forma predeterminada al valor `##t` (vertader) per a tots els pentagrames. Si s’estableix a `##f` (fals) en un context de **Staff** determinat s’aplicarà solament a les alteracions d’aquest pentagrama. Si s’estableix a fals en el context de la partitura, **Score**, s’aplicarà a tots els pentagrames.

Així, això desactivarà els bequadres addicionals a un pentagrama:

```
<<
\new Staff \relative c'' {
  aeses2 aes
}
\new Staff \relative c'' {
  \set Staff.extraNatural = ##f
  aeses2 aes
}
>>
```



i això els desactivarà a tots els pentagrames:

```
<<
  \new Staff \relative c'' {
    aeses2 aes
  }
  \new Staff \relative c'' {
    \set Score.extraNatural = ##f
    aeses2 aes
  }
>>
```



Com un exemple més, si s'estableix `clefTransposition` dins del context de `Score`, aquesta instrucció canvia immediatament el valor de la transposició en tots els pentagrames en curs i estableix un nou valor predeterminat que s'aplicarà a tots els pentagrames.

La instrucció oposada, `\unset`, té l'efecte de suprimir la propietat del context, el que ocasiona que la major part de les propietats tornin al seu valor predeterminat. Normalment no és necessari l'ús de `\unset`, atès que una nova instrucció `\set` farà l'ajust desitjat.

Les instruccions `\set` i `\unset` poden aparèixer en qualsevol lloc del fitxer d'entrada i tindran un efecte a partir del temps on es troben i fins al final de la partitura o fins que la propietat es torni a establir mitjançant `\set` o `\unset`. Provem a modificar la mida de la font tipogràfica, el que afecta la mida dels caps de les notes (entre altres coses) diverses vegades. El canvi s'agafa a partir del valor predeterminat, no el valor en curs.

```
c4 d
% fes que els caps de nota siguin més petits
\set fontSize = #-4
e4 f |
% fes que els caps de nota siguin més grans
\set fontSize = #2.5
g4 a
% torna a la mida predeterminada
\unset fontSize
b4 c |
```



Hem pogut veure com establir els valors de diversos tipus de propietat diferents. Observeu que els nombres enters i reals van sempre precedits d'un símbol de coixinet, `#`, mentre que un

valor booleà vertader o fals s'especifica mitjançant **##t** i **##f**, amb dos coixinets. Una propietat de test s'ha de tancar entre cometes dobles, com abans, tot i que veurem més endavant que el text realment es pot especificar d'una forma molt més general utilitzant la molt potent instrucció **markup**.

Canviar les propietats d'un context amb **\with**

El valor predeterminat de les propietats de context es pot establir en el moment que es crea el context. A vegades aquesta forma d'establir el valor d'una propietat és molt més clara, si ha de quedar fix durant tot el temps que duri el context. Quan es crea un context amb una instrucció **\new** pot anar immediatament seguit d'un bloc **\with { ... }** en el que s'estableix els valors predeterminats de les propietats. Per exemple, si volem suprimir la impressió de bequadres addicionals per a tota la duració d'un pentagrama, podem escriure:

```
\new Staff \with { extraNatural = ##f }
```

de la forma següent:

```
<<
  \new Staff {
    \relative c'' {
      gisis4 gis aeses aes
    }
  }
  \new Staff \with { extraNatural = ##f } {
    \relative c'' {
      gisis4 gis aeses aes
    }
  }
>>
```



Les propietats ajustades d'aquesta manera encara poden canviar-se dinàmicament utilitzant **\set** i tornar-se al valor predeterminat que es va establir al bloc **\with** mitjançant **\unset**.

Així doncs, si la propietat **fontSize** s'ajusta dins d'una instrucció **\with**, té l'efecte de reiniciar el valor predeterminat de la mida de la font tipogràfica. Si més tard es modifica amb **\set**, aquest nou valor predeterminat pot restablir-se amb la instrucció **\unset fontSize**.

Canviar las propietats d'un context amb **\context**

Els valors de propietat dels contextos es poden establir per a *tots* els contextos d'un tipus determinat, com per exemple tots els contextos de **Staff**, amb una única instrucció. El tipus de context s'identifica mitjançant la utilització del nom del seu tipus, com **Staff**, precedit d'una barra invertida: **\Staff**. L'enunciat que estableix el valor de la propietat és el mateix que el que està en un bloc **\with**, presentat anteriorment. Es col·loca en un bloc **\context** dins d'un bloc **\layout**. Cada bloc **\context** afecta a tots els contextos del tipus especificat al llarg del bloc **\score** o **\book** en el que apareix el bloc **\layout**. A continuació presentem un exemple que mostra el format:

```
\score {
```

```

\new Staff {
  \relative c'' {
    cisis4 e d cis
  }
}
\layout {
  \context {
    \Staff
    extraNatural = ##t
  }
}

```



Si es vol aplicar la sobreescritura de propietats a tots els pentagrames de la partitura:

```

\score {
  <<
    \new Staff {
      \relative c'' {
        gisis4 gis aeses aes
      }
    }
    \new Staff {
      \relative c'' {
        gisis4 gis aeses aes
      }
    }
  >>
  \layout {
    \context {
      \Score extraNatural = ##f
    }
  }
}

```



Les propietats de context establertes d'aquesta forma es poden sobreescriure per a exemples concrets de contextos mitjançant enunciats dins d'un bloc `\with`, i mitjançant instruccions `\set` intercalades dins d'enunciats musicals.

Vegeu també

Referència de la notació: Secció “Changing context default settings” in *Referència de la notació*. Secció “The set command” in *Referència de la notació*

Referència de funcionament intern: *Secció “Contexts” in Referència de funcionament intern*, *Secció “Tunable context properties” in Referència de funcionament intern*.

3.3.5 Afegir i eliminar gravadors

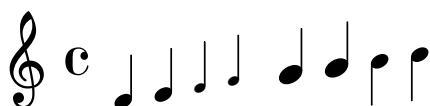
Hem vist que cada un dels contextos conté diversos gravadors, cada u dels quals és al seu cop responsable de la producció d’una fracció particular del resultat imprès, com ara línies divisòries, pentagrames, caps, pliques, etc. Si un gravador és eliminat d’un context, ja no podrà produir la seva sortida impresa. És una forma una mica radical de modificar la sortida, però algunes vegades pot ser útil.

Canviar un sol context

Per eliminar un gravador d’un context únic, usem la instrucció `\with` situada immediatament després de la instrucció que crea el context, com a la secció anterior.

Com a il·lustració, repetim un exemple estret de la secció anterior amb les línies del pentagrama eliminades. Recordeu que les línies del pentagrama estan dibuixades pel gravador `Staff_symbol_engraver`.

```
\new Staff \with {
  \remove "Staff_symbol_engraver"
}
\relative c' {
  c4 d
  \set fontSize = #-4 % fes més petits els caps de les notes
  e4 f |
  \set fontSize = #2.5 % fes més grans els caps de les notes
  g4 a
  \unset fontSize % retorna a la mida predeterminada
  b4 c |
}
```



Els gravadors també es poden afegir als contextos individuals. La instrucció que ho fa és

`\consists Nom_del_gravador`,

situada dins d’un bloc `\with`. Certes partitures vocals tenen una indicació d’àmbit o tessitura situada al principi del pentagrama per indicar l’àmbit de notes en aquest pentagrama, vegeu *Secció “ambitus” in Glossari musical*. L’ambitus es produeix per part del gravador `Ambitus_engraver`, que normalment no està inclòs en cap context. Si l’afegim al context `Voice`, calcula el rang a partir d’aquesta única veu:

```
\new Staff <<
  \new Voice \with {
    \consists "Ambitus_engraver"
  } {
    \relative c'' {
      \voiceOne
      c4 a b g
    }
  }
  \new Voice {
    \relative c' {
```

```

\voiceTwo
c4 e d f
}
}
>>

```



però si afegim el gravador d'àmbit al context de **Staff**, calcula el rang de totes les notes en totes les veus d'aquest pentagrama:

```

\new Staff \with {
  \consists "Ambitus_engraver"
}
<<
  \new Voice {
    \relative c'' {
      \voiceOne
      c4 a b g
    }
  }
  \new Voice {
    \relative c' {
      \voiceTwo
      c4 e d f
    }
  }
}
>>

```



Canviar tots els contextos del mateix tipus

Els exemples anteriors mostren la manera d'eliminar o afegir gravadors als contextos individuals. També és possible eliminar o afegir gravadors a tots els contextos d'un tipus específic, situant les instruccions al context corresponent dins d'un bloc **\layout**. Per exemple, si volem mostrar els rangs de tessitura per a tots els pentagrames d'una partitura de quatre pautes, podem escriure

```

\score {
  <<
    \new Staff {
      \relative c'' {
        c4 a b g
      }
    }
    \new Staff {
      \relative c' {
        c4 a b g
      }
    }
  }
}

```

```

    }
    \new Staff {
      \clef "G_8"
      \relative c' {
        c4 a b g
      }
    }
    \new Staff {
      \clef "bass"
      \relative c {
        c4 a b g
      }
    }
  }
  >>
  \layout {
    \context {
      \Staff
      \consists "Ambitus_engraver"
    }
  }
}

```



Els valors predeterminats de les propietats dels contextos també es poden establir per a tots els contextos d'un tipus en particular incloent-hi la instrucció `\set` dins d'un bloc `\context` de la mateixa forma.

Vegeu també

Referència de la notació: [Secció “Modifying context plug-ins”](#) in *Referència de la notació*, [Secció “Changing context default settings”](#) in *Referència de la notació*.

Advertiments i problemes coneguts

Els gravadors `Stem_engraver` i `Beam_engraver` (de plica i de barra) adjunten al cap de les notes els objectes que creen. Si es suprimeix el gravador de caps de nota `Note_heads_engraver`, no es produeix cap cap i per tant no es creen tampoc pliques ni barres.

3.4 Extensió de les plantilles

Heu llegit el tutorial i ara sabeu escriure música. Però, com podeu posar els pentagrames que voleu? Les plantilles estan molt bé, però què passa si voleu alguna cosa que està en cap plantilla?

Bé, podeu trobar muntanyes de plantilles (vegeu [Secció “Plantilles”](#) in *Manual d’aprenentatge*) que us poden servir com a punt de partida. Però i si voleu quelcom que no està contemplat aquí? Continueu llegint.

3.4.1 Soprano i violoncel

Per començar, agafeu la plantilla que us sembli més semblant a allò que voleu aconseguir. Diguem-ne que voleu escriure alguna cosa per a soprano i violoncel. En aquest cas començaríem amb la plantilla ‘Notes i lletra’ (per a la part de soprano).

```
\version "2.19.16"
melodia = \relative c' {
  \clef "treble"
  \key c \major
  \time 4/4
  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score {
  <<
    \new Voice = "u" {
      \autoBeamOff
      \melodia
    }
    \new Lyrics \lyricsto "u" \text
  >>
  \layout { }
  \midi { }
}
```

Ara volem afegir una part de violoncel. Vegem l'exemple ‘Sols notes’:

```
\version "2.19.16"
melodia = \relative c' {
  \clef "treble"
  \key c \major
  \time 4/4
  a4 b c d
}

\score {
  \new Staff \melodia
  \layout { }
  \midi { }
}
```

No necessitem dues instruccions `\version`. Ens caldrà la secció `melodia`. No volem dues seccions `\score` (si tinguéssim dues `\scores`, acabariem amb dues partícels per separat.) Volem les dues juntes, com un duo. Dins de la secció `\score`, no ens fan falta dos `\layout` ni dos `\midi`.

Si ens limitéssim a copiar i enganxar la secció `melodia`, acabariem amb dues seccions `melodia` separades, així que anem a canviar-los el nom. Anomenarem `musicaSoprano` a la secció de la

soprano i `musicaVioloncel` a la secció de violoncel. Al mateix temps canviarem el nom de `text` a `lletraSoprano`. Recordeu canviar el nom a les dues aparicions de totes aquests noms – tant la definició inicial (la part `melodia = relative c' {`) – com l'ús d'aquest nom (en la secció `\score`).

També aprofitarem per canviar el pentagrama de la part del violoncel (els violoncels s'escriuen normalment en clau de Fa). Així mateix, canviarem algunes del violoncel.

```
\version "2.19.16"
musicaSoprano = \relative c' {
  \clef "treble"
  \key c \major
  \time 4/4
  a4 b c d
}

lletraSoprano = \lyricmode {
  Aaa Bee Cee Dee
}

musicaVioloncel = \relative c {
  \clef "bass"
  \key c \major
  \time 4/4
  d4 g fis8 e d4
}

\score{
  <<
    \new Voice = "u" {
      \autoBeamOff
      \musicaSoprano
    }
    \new Lyrics \lyricsto "u" \lletraSoprano
  >>
  \layout { }
  \midi { }
}
```

Això té una aparença prometedora, però la part del violoncel no surt a la partitura (no l'hem posada a la secció `\score`). Si volem que la part del violoncel aparegui a sota de la de soprano, hem d'afegir

```
\new Staff \musicaVioloncel
```

just a sota de tot el codi de la soprano. També hem de posar `<< i >>` abans i després de la música – el que indica al LilyPond que hi ha més d'una cosa (en aquest cas, **Staff**) succeint al mateix moment –. La `\score` s'assemblarà ara a això:

```
\score {
  <<
  <<
    \new Voice = "u" {
      \autoBeamOff
      \musicaSoprano
    }
  >>
  >>
}
```

```

    \new Lyrics \lyricsto "u" \llettraSoprano
  >>
  \new Staff \musicaVioloncel
  >>
  \layout { }
  \midi { }
}

```

Això sembla una mica enrevessat; el marges estan desquadrats. Això té fàcil solució. Presentem aquí la plantilla completa per a soprano i violoncel.

```

\version "2.19.16"

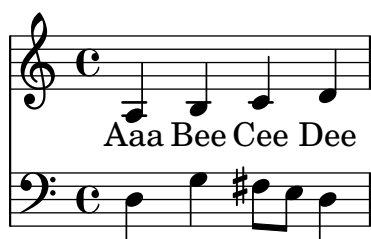
musicaSoprano = \relative c' {
  \clef "treble"
  \key c \major
  \time 4/4
  a4 b c d
}

llettraSoprano = \lyricmode {
  Aaa Bee Cee Dee
}

musicaVioloncel = \relative c {
  \clef "bass"
  \key c \major
  \time 4/4
  d4 g fis8 e d4
}

\score {
  <<
    <<
      \new Voice = "u" {
        \autoBeamOff
        \musicaSoprano
      }
      \new Lyrics \lyricsto "u" \llettraSoprano
    >>
    \new Staff \musicaVioloncel
  >>
  \layout { }
  \midi { }
}

```



Vegeu també

Les plantilles d'inici es poden trobar a l'apèndix 'Plantilles', vegeu [Secció "Plantilles de pentagrama únic" in *Manual d'aprenentatge*](#).

3.4.2 Partitura vocal a quatre veus SATB

La major part de les partitures vocals escrites per a cor mixt a quatre veus amb acompanyament orquestral, com l'«Elies» de Mendelssohn o el «Messies» de Haendel, tenen la música coral i la lletra en quatre pentagrames S, A, T i B, respectivament, amb una reducció de piano de l'acompanyament d'orquestra, per sota. Heus aquí un exemple del «Messies» de Haendel:

Cap de les plantilles proporciona aquesta disposició amb exactitud. La més semblant és [Secció "Partitura vocal SATB i reducció per a piano automàtica" in *Manual d'aprenentatge*](#), però necessitem canviar la disposició i afegir un acompanyament de piano que no estigui derivat automàticament de les parts vocals. Les variables que contenen la música i la lletra de les parts vocals és adequat, però haurem d'afegir variables per a la reducció de piano.

L'ordre en què apareixen els contextos al ChoirStaff de la plantilla no es correspon amb l'ordre de la partitura vocal que hem mostrat més amunt. Hem de reordenar-los perquè hi hagi quatre pentagrames amb la lletra escrita directament a sota de les notes de cada part. Totes les veus han de ser `\voiceOne`, que és la predeterminada, perquè les instruccions `\voiceXXX` es puguin eliminar. També hem d'especificar la clau de tenor (clau de sol octava baixa) a les parts de tenor. Encara no hem trobat la forma que la lletra s'especifica a la plantilla, així que hem d'utilitzar el mètode que ens resulta familiar. També hem d'escriure els noms de cada pentagrama.

En fer-lo així obtenim el ChoirStaff següent:

```
\new ChoirStaff <<
  \new Staff = "sopranos" <<
    \set Staff.instrumentName = #"Soprano"
    \new Voice = "sopranos" {
```

```

        \global
        \musicaSoprano
    }
>>
\new Lyrics \lyricsto "sopranos" {
    \llettraSoprano
}
\new Staff = "altos" <<
    \set Staff.instrumentName = #"Alto"
    \new Voice = "altos" {
        \global
        \musicaAlto
    }
>>
\new Lyrics \lyricsto "altos" {
    \llettraAlto
}
\new Staff = "tenors" <<
    \set Staff.instrumentName = #"Tenor"
    \new Voice = "tenors" {
        \global
        \musicaTenor
    }
>>
\new Lyrics \lyricsto "tenors" {
    \llettraTenor
}
\new Staff = "baixos" <<
    \set Staff.instrumentName = #"Baix"
    \new Voice = "baixos" {
        \global
        \musicaBaix
    }
>>
\new Lyrics \lyricsto "baixos" {
    \llettraBaix
}
>> % fi del ChoirStaff

```

A continuació podem treballar sobre la part de piano. És fàcil: tan sols s'ha de treure la part de piano de la plantilla de 'Piano solista':

```

\new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff = "superior" \superior
    \new Staff = "inferior" \inferior
>>

```

i escriure les definicions de variable per a **superior** i **inferior**.

Els grups **ChoirStaff** i **PianoStaff** s'han de combinar utilitzant angles dobles, atès que els volem apilar l'un sobre l'altre

```

<< % combina els grups ChoirStaff i PianoStaff l'un sobre l'altre
\new ChoirStaff <<
    \new Staff = "sopranos" <<

```



```

        \new Voice = "sopranos" {
            \global
            \musicaSoprano
        }
>>
\new Lyrics \lyricsto "sopranos" {
    \llettraSoprano
}
\new Staff = "altos" <<
    \new Voice = "altos" {
        \global
        \musicaAlto
    }
>>
\new Lyrics \lyricsto "altos" {
    \llettraAlto
}
\new Staff = "tenores" <<
    \clef "G_8" % clave de tenor
    \new Voice = "tenores" {
        \global
        \musicaTenor
    }
>>
\new Lyrics \lyricsto "tenores" {
    \llettraTenor
}
\new Staff = "baixos" <<
    \clef "bass"
    \new Voice = "baixos" {
        \global
        \musicaBaix
    }
>>
\new Lyrics \lyricsto "baixos" {
    \llettraBaix
}
>> % fi del ChoirStaff

\new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano"
    \new Staff = "upper" \upper
    \new Staff = "lower" \lower
>>
>>

```

En combinar tot això junt i escriure la música dels tres compassos de l'exemple anterior, obtenim:

```

\version "2.19.16"
global = { \key d \major \time 4/4 }
musicaSoprano = \relative c' {
    \clef "treble"

```

```

    r4 d2 a4 | d4. d8 a2 | cis4 d cis2 |
}
llettraSoprano = \lyricmode {
    Wor -- thy | is the lamb | that was slain |
}
musicaAlto = \relative a' {
    \clef "treble"
    r4 a2 a4 | fis4. fis8 a2 | g4 fis fis2 |
}
llettraAlto = \llettraSoprano
musicaTenor = \relative c' {
    \clef "G_8"
    r4 fis2 e4 | d4. d8 d2 | e4 a, cis2 |
}
llettraTenor = \llettraSoprano
musicaBaix = \relative c' {
    \clef "bass"
    r4 d2 cis4 | b4. b8 fis2 | e4 d a'2 |
}
llettraBaix = \llettraSoprano
upper = \relative a' {
    \clef "treble"
    \global
    r4 <a d fis>2 <a e' a>4 |
    <d fis d'>4. <d fis d'>8 <a d a'>2 |
    <g cis g'>4 <a d fis> <a cis e>2 |
}
lower = \relative c, {
    \clef "bass"
    \global
    <d d'>4 <d d'>2 <cis cis'>4 |
    <b b'>4. <b' b'>8 <fis fis'>2 |
    <e e'>4 <d d'> <a' a'>2 |
}

\score {
  << % combina el ChoirStaff i PianoStaff en paral.lel
  \new ChoirStaff <<
    \new Staff = "sopranos" <<
      \set Staff.instrumentName = #"Soprano"
      \new Voice = "sopranos" {
        \global
        \musicaSoprano
      }
    >>
    \new Lyrics \lyricsto "sopranos" {
      \llettraSoprano
    }
  \new Staff = "altos" <<
    \set Staff.instrumentName = #"Alto"
    \new Voice = "altos" {
      \global

```

```

        \musicaAlto
    }
>>
\new Lyrics \lyricsto "altos" {
    \llettraAlto
}
\new Staff = "tenors" <<
    \set Staff.instrumentName = #"Tenor"
    \new Voice = "tenors" {
        \global
        \musicaTenor
    }
>>
\new Lyrics \lyricsto "tenors" {
    \llettraTenor
}
\new Staff = "baixos" <<
    \set Staff.instrumentName = #"Baix"
    \new Voice = "baixos" {
        \global
        \musicaBaix
    }
>>
\new Lyrics \lyricsto "baixos" {
    \llettraBaix
}
>> % fi ChoirStaff

\new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano  "
    \new Staff = "upper" \upper
    \new Staff = "lower" \lower
>>
>>
}
```

The image shows a musical score for five parts: Soprano, Alto, Tenor, Baix (Bass), and Piano. The lyrics for the vocal parts are "Worthy is the lamb that was slain". The score is written in G major (one sharp) and common time (C). The vocal parts are in treble clef (Soprano, Alto, Tenor) and bass clef (Baix). The Piano part is in grand staff (treble and bass clefs). The music is a simple setting of the text, with each vocal part having a single line of music and the piano part having a single line of music.

3.4.3 Crear una partitura partint de zero

Després d'adquirir una mica de soltesa en l'escriptura del codi del LilyPond, us adonareu que és més fàcil construir completament una partitura partint de zero, que modificar una plantilla. També podeu desenvolupar el vostre propi estil de forma que s'adapti al tipus de música que us agradi. Vegem a continuació com confeccionar una partitura per a un preludi d'òrgan, com a exemple.

Comencem amb una secció per al encapçalament. Aquí és on van el títol, nom del compositor, etc., després van les definicions de les variables, i finalment el bloc de partitura. Comencem a veure-les per sobre i més tard completarem els detalls.

Utilitzarem els dos primers compassos del preludi de Bach basat en *Jesu, meine Freude*, que està escrit per a òrgan amb dos manuals i pedal. Conté els dos compassos següents de música al final de la secció. La part del manual superior té dues veus, i l'inferior i el pedal, una veu cada u. Així doncs, necessitem quatre definicions per a la música i una més per definir el compàs i la tonalitat:

```
\version "2.19.16"
\header {
  title = "Jesu, meine Freude"
  composer = "J S Bach"
}
keyTime = { \key c \minor \time 4/4 }
MusicaManualUVeuU = { s1 }
MusicaManualUVeuDos = { s1 }
MusicaManualDos = { s1 }
MusicaOrganPedal = { s1 }

\score {
}
```

De moment hem escrit tan sols una nota espaiadora, **s1**, en lloc de la música de veritat. Li afegirem més endavant.

A continuació vegem què va al bloc de partitura. Senzillament, reflectirem l'estructura de pentagrames que desitgem. La música d'òrgan s'escriu en general en tres pentagrames, un per cada un dels manuals i l'altre pel pedal. Els pentagrames dels manuals s'abasten amb una clau, així que els inclourem en un grup PianoStaff. La primera part de manual té dues veus, i la segona una sola.

```
\new PianoStaff <<
  \new Staff = "ManualU" <<
    \new Voice {
      \MusicaManualUVeuU
    }
    \new Voice {
      \MusicaManualUVeuDos
    }
  >> % fi del context de Staff ManualU
  \new Staff = "ManualDos" <<
    \new Voice {
      \MusicaManualDos
    }
  >> % fi del context de Staff ManualDos
>> % fi del context de PianoStaff
```

Després, hem d'afegir un pentagrama per a l'òrgan de pedal. Això va per sota del PianoStaff, però ha de ser simultani amb ell, per la qual cosa escrivim angles dobles rodejant als dos. Si ens oblidem d'això, es produirà un error al fitxer log de registre. És un error molt comú que cometrà abans o després! Intenteu copiar l'exemple final que apareix al final de la secció, esborreu els dobles angles i processeu el fitxer per veure quin error produeix.

```
<< % el grup PianoStaff i el pentagrama de Pedal son simultanis
  \new PianoStaff <<
    \new Staff = "ManualU" <<
      \new Voice {
        \MusicaManualUVeuU
      }
      \new Voice {
        \MusicaManualUVeuDos
      }
    >> % fi del context de Staff ManualU
    \new Staff = "ManualDos" <<
      \new Voice {
        \MusicaManualDos
      }
    >> % fi del context de Staff ManualDos
  >> % fi del context de PianoStaff
  \new Staff = "OrganPedal" <<
    \new Voice {
      \MusicaOrganPedal
    }
  >>
>>
```

No és necessari utilitzar la construcció simultània << ... >> per al pentagrama del manual dos i el pentagrama del òrgan de pedal, atès que contenen una única expressió, però no fa mal, i és un bon costum utilitzar sempre dobles angles després de `\new Staff` quan hi diverses veus.

El contrari és cert per a les veus: normalment han d'anar seguides de claudàtors { ... } en cas que tinguem música codificada com a variables diferents que s'han de situar consecutivament.

Afegim aquesta estructura al bloc de partitura, i ajustem el sagnat dels marges. També escrivim els claudàtors corresponents, ens assegurem que les pliques i lligadures d'unió i expressió en cada una de les veus del pentagrama superior apunten en la direcció adequada amb `\voiceOne` i `\voiceTwo` i escrivim el compàs i la tonalitat en cadascú dels pentagrames usant la nostra variable prèviament definida `\TimeKey`.

```
\score {
  << % el grup PianoStaff i el pentagrama de Pedal són simultanis
  \new PianoStaff <<
    \new Staff = "ManualU" <<
      \keyTime % establir compàs i tonalitat
      \clef "treble"
      \new Voice {
        \voiceOne
        \MusicaManualUVeuU
      }
      \new Voice {
        \voiceTwo
        \MusicaManualUVeuDos
      }
    >> % fi del context de Staff ManualU
  \new Staff = "ManualDos" <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaManualDos
    }
    >> % fi del context de Staff ManualDos
  >> % fi del context de PianoStaff
  \new Staff = "OrganPedal" <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaOrganPedal
    }
    >> % fi del pentagrama de OrganPedal
  >>
} % fi del context Score
```

La disposició anterior dels pentagrames d'òrgan és gairebé perfecta; tanmateix, hi ha un lleuger defecte que no és visible quan s'observa un sol sistema: la distància entre el pentagrama de pedal i el de la mà esquerra hauria de ser aproximadament la mateixa que la que hi ha entre els pentagrames de les mans esquerra i dreta. Concretament, la ampliabilitat dels pentagrames dins d'un context `PianoStaff` és limitada (de forme que la distància entre els pentagrames de les mans esquerra i dreta mai no creixin excessivament), i el pentagrama dels pedals hauria de comportar-se de una manera semblant.

El grau d'ampliabilitat i separabilitat dels pentagrames es pot controlar amb la propietat `staff-staff-spacing` de l'objecte gràfic `VerticalAxisGroup` (els objectes gràfics reben en general el nom de 'grob's a la documentació del LilyPond); no us preocupeu de moment dels detalls, ja que això s'explica més tard de forma exhaustiva. Els més curiosos podeu donar una ullada a [Secció "Overview of modifying properties" in Referència de la notació](#). En aquest cas

volem modificar solament la sub-propietat `stretchability`. Un altre cop, els curiosos trobareu els valors predeterminats per a la propietat `staff-staff-spacing` al fitxer `'scm/define-grobs.scm'` examinant la definició del grob `VerticalAxisGroup`. El valor de `stretchability` s'agafa de la definició del context `PianoStaff` (al fitxer `'ly/engraver-init.ly'`) de forma que els valors siguin idèntics.

```
\score {
  << % el grup PianoStaff i el pentagrama de Pedal son simultanis
  \new PianoStaff <<
    \new Staff = "ManualU" <<
      \keyTime % establir compàs y tonalitat
      \clef "treble"
      \new Voice {
        \voiceOne
        \MusicaManualUVeuU
      }
      \new Voice {
        \voiceTwo
        \MusicaManualUVeuDos
      }
    >> % fi del context de Staff ManualU
  \new Staff = "ManualDos" \with {
    \override VerticalAxisGroup.staff-staff-spacing.stretchability = 5
  } <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaManualDos
    }
  >> % fi del context de Staff ManualDos
  >> % fi del context de PianoStaff
  \new Staff = "OrganPedal" <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaOrganPedal
    }
  >> % fi del pentagrama de OrganPedal
  >>
} % fi del context Score
```

Amb això es completa l'estructura. Tota música per a òrgan de tres pentagrames tindrà una estructura semblant, tot i que el nombre de veus pot variar. Tot el que ens queda és afegir la música, i combinar totes les parts.

```
\version "2.19.16"
\header {
  title = "Jesu, meine Freude"
  composer = "J S Bach"
}
keyTime = { \key c \minor \time 4/4 }
MusicaManualUVeuU = \relative g' {
  g4 g f ees |
  d2 c |
```

```

}
MusicaManualUveuDos = \relative c' {
  ees16 d ees8~ 16 f ees d c8 d~ d c~ |
  8 c4 b8 c8. g16 c b c d |
}
MusicaManualDos= \relative c' {
  c16 b c8~ 16 b c g a8 g~ 16 g aes ees |
  f16 ees f d g aes g f ees d ees8~ 16 f ees d |
}
MusicaOrganPedal = \relative c {
  r8 c16 d ees d ees8~ 16 a, b g c b c8 |
  r16 g ees f g f g8 c,2 |
}

\score {
  << % PianoStaff i Pedal Staff han de ser simultanis
  \new PianoStaff <<
    \new Staff = "ManualU" <<
      \keyTime % establir la clau i l'armadura
      \clef "treble"
      \new Voice {
        \voiceOne
        \MusicaManualUveuU
      }
      \new Voice {
        \voiceTwo
        \MusicaManualUveuDos
      }
    >> % fi del context Staff ManualU
  \new Staff = "ManualDos" \with {
    \override VerticalAxisGroup.staff-staff-spacing.stretchability = 5
  } <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaManualDos
    }
  >> % fi del context Staff ManualDos
  >> % fi del context PianoStaff
  \new Staff = "PedalOrgan" <<
    \keyTime
    \clef "bass"
    \new Voice {
      \MusicaOrganPedal
    }
  >> % fi del context Staff PedalOrgan
  >>
} % fi del context Score

```

Jesu, meine Freude

J S Bach



Vegeu també

Glossari musical: *Secció “system” in Glossari musical.*

3.4.4 Estalviar tecleig mitjançant variables i funcions

Arribats a aquest punt, heu vist coses d'aquest tipus:

```
hornNotes = \relative c'' { c4 b dis c }
```

```
\score {
  {
    \hornNotes
  }
}
```



Fins i tot us adonareu que això pot ser útil en música minimalista:

```
fragmentA = \relative c'' { a4 a8. b16 }
fragmentB = \relative c'' { a8. gis16 ees4 }
```

```
violí = \new Staff {
  \fragmentA \fragmentA |
  \fragmentB \fragmentA |
}
```

```
\score {
  {
    \violí
  }
}
```



Tot i així també es pot fer servir aquests identificadors (que també es coneixen com a variables, macros o instruccions definides per l'usuari) per fer trucs:

```
dolce = \markup { \italic \bold dolce }

padText = { \once \override TextScript.padding = #5.0 }
fthenp = _\markup {
  \dynamic f \italic \small { 2nd } \hspace #0.1 \dynamic p
}

violí = \relative c'' {
  \repeat volta 2 {
    c4._\dolce b8 a8 g a b |
    \padText
    c4.^"hi there!" d8 e' f g d |
    c,4.\fthenp b8 c4 c-. |
  }
}

\score {
{
  \violí
}
\layout { ragged-right = ##t }
}
```



Òbviament aquests identificadors són útils per estalviar tecleig. Però són dignes de tenir en compte fins i tot si s'utilitzaran un sol cop: redueixen la complexitat. Examinem l'exemple anterior reescrit sense cap identificador. Trobareu que és molt més difícil de llegir, sobretot l'última línia.

```
violí = \relative c'' {
  \repeat volta 2 {
    c4._\markup { \italic \bold dolce } b8 a8 g a b |
    \once \override TextScript.padding = #5.0
    c4.^"hi there!" d8 e' f g d |
    c,4.\markup {
      \dynamic f \italic \small { 2nd } \hspace #0.1 \dynamic p
    }
    b8 c4 c-. |
  }
}
```

Fins ara hem contemplat la substitució estàtica: quan el LilyPond es troba amb `\padText`, el substitueix amb allò que hem definit que sigui (és a dir, tot el que està a la dreta de `padtext=`).

El LilyPond també pot gestionar substitucions no estàtiques (penseu en elles com funcions).

```

padText =
#(define-music-function
  (parser location padding)
  (number?)
  #{
    \once \override TextScript.padding = #padding
  #})

\relative c''' {
  c4^"piu mosso" b a b |
  \padText #1.8
  c4^"piu mosso" d e f |
  \padText #2.6
  c4^"piu mosso" fis a g |
}

```



La utilització d'identificadors també és una bona forma de reduir el treball si la sintaxis d'entrada del LilyPond canvia (vegeu [Secció “Actualització de fitxers amb convert-ly” in *Utilització del programa*](#)). Si teniu una sola definició (com ara `\dolce`) per a tots els fitxers (vegeu [Secció “Fulls d'estil” in *Manual d'aprenentatge*](#)), i després la sintaxis es modifica, sols haurà d'actualitzar la seva definició `\dolce` única, enlloc de haver de fer canvis a cadascú dels fitxers `‘.ly’`.

3.4.5 Partitures i partícels

En música orquestral, totes les notes s'imprimeixen dues vegades. Un cop en les partícels per a tots els músics, i una altra per a la partitura del director. Els identificadors es poden usar per evitar la duplicació del treball. La música s'escriu un cop s'emmagatzema en una variable. El contingut d'aquesta variable s'usa després per generar tant la partícula com la partitura del director.

És molt convenient definir les notes en un fitxer especial. Per exemple, suposem que el fitxer `‘trompa.ly’` conté la següent part d'un duo per a trompa i fagot:

```

notesTrompa = \relative c {
  \time 2/4
  r4 f8 a | cis4 f | e4 d |
}

```

Després es fa una partícula escrivint en un fitxer el següent:

```

\include "trompa.ly"

\header {
  instrument = "Trompa en Fa"
}

{
  \transpose f c' \notesTrompa
}

```

La línia

```
\include "trompa.ly"
```

substitueix el contingut de 'trompa.ly' en aquesta posició dins del fitxer, així que `notesTrompa` es defineix amb posterioritat. La instrucció `\transpose f c'` indica que l'argument constituït per `notesTrompa` s'ha de transposar una quina cap amunt. El que sona com `f` s'escriu com `c'`, el que correspon amb el to d'afinació d'una trompa normal en Fa. La transposició es pot veure a la següent sortida



A peces per a conjunt, amb freqüència una de les veus no sona durant molts compassos. Això queda denotat per un silenci especial, el silenci multicompass. S'introdueix amb una `R` majúscula seguida d'una duració (1 en el cas de la rodona, 2 en el caso de una blanca, etc.). Multiplicant la duració es poden construir silencis més llargs. Per exemple, aquest silenci ocupa 3 compassos de 2/4

```
R2*3
```

Quan s'imprimeix la partitura s'han de comprimir els silencis multicompass. Això es fa establint una variable de temps de execució

```
\set Score.skipBars = ##t
```

Aquesta instrucció estableix el valor de la propietat `skipBars` al context de `Score` a vertader (`##t`). Anteposant el silenci i aquesta opció a la música anterior, arribem al següent resultat



Aquesta partitura es fa combinant tota la música junta. Suposant que l'altra veu es troba dins de `notesFagot` al fitxer 'fagot.ly', la partitura es fa amb

```
\include "fagot.ly"
\include "trompa.ly"
```

```
<<
  \new Staff \notesTrompa
  \new Staff \notesFagot
>>
```

el que ens porta a



4 Tweaking output

This chapter discusses how to modify output. LilyPond is extremely configurable; virtually every fragment of output may be changed.

4.1 Tweaking basics

4.1.1 Introduction to tweaks

‘Tweaking’ is a LilyPond term for the various methods available to the user for modifying the actions taken during interpretation of the input file and modifying the appearance of the printed output. Some tweaks are very easy to use; others are more complex. But taken together the methods available for tweaking permit almost any desired appearance of the printed music to be achieved.

In this section we cover the basic concepts required to understand tweaking. Later we give a variety of ready-made commands which can simply be copied to obtain the same effect in your own scores, and at the same time we show how these commands may be constructed so that you may learn how to develop your own tweaks.

Before starting on this Chapter you may wish to review the section [Contexts and engravers](#), as Contexts, Engravers, and the Properties contained within them are fundamental to understanding and constructing Tweaks.

4.1.2 Objects and interfaces

Tweaking involves modifying the internal operation and structures of the LilyPond program, so we must first introduce some terms which are used to describe those internal operations and structures.

The term ‘Object’ is a generic term used to refer to the multitude of internal structures built by LilyPond during the processing of an input file. So when a command like `\new Staff` is encountered a new object of type `Staff` is constructed. That `Staff` object then holds all the properties associated with that particular staff, for example, its name and its key signature, together with details of the engravers which have been assigned to operate within that staff’s context. Similarly, there are objects to hold the properties of all other contexts, such as `Voice` objects, `Score` objects, `Lyrics` objects, as well as objects to represent all notational elements such as bar lines, note heads, ties, dynamics, etc. Every object has its own set of property values.

Some types of object are given special names. Objects which represent items of notation on the printed output such as note heads, stems, slurs, ties, fingering, clefs, etc are called ‘Layout objects’, often known as ‘Graphical Objects’, or ‘Grobs’ for short. These are still objects in the generic sense above, and so they too all have properties associated with them, such as their position, size, color, etc.

Some layout objects are still more specialized. Phrasing slurs, crescendo hairpins, ottava marks, and many other grobs are not localized in a single place – they have a starting point, an ending point, and maybe other properties concerned with their shape. Objects with an extended shape like these are called ‘Spanners’.

What is more, there are ‘abstract’ grobs which do not print anything of their own, but rather collect, position and manage other grobs. Common examples for this are `DynamicLineSpanner`, `BreakAlignment`, `NoteColumn`, `VerticalAxisGroup`, `NonMusicalPaperColumn` and similar. We will see how some of these are used later.

It remains to explain what ‘Interfaces’ are. Many objects, even though they are quite different, share common features which need to be processed in the same way. For example, all grobs have a color, a size, a position, etc, and all these properties are processed in the same way during LilyPond’s interpretation of the input file. To simplify these internal operations these

common actions and properties are grouped together in an object called a **grob-interface**. There are many other groupings of common properties like this, each one given a name ending in **interface**. In total there are over 100 such interfaces. We shall see later why this is of interest and use to the user.

These, then, are the main terms relating to objects which we shall use in this chapter.

4.1.3 Naming conventions of objects and properties

We met some object naming conventions previously, in [\[Contexts and engravers\]](#), [pàgina \[\\[Modifying context properties\\]\]\(#\)](#). Here for reference is a list of the most common object and property types together with the conventions for naming them and a couple of examples of some real names. We have used ‘A’ to stand for any capitalized alphabetic character and ‘aaa’ to stand for any number of lower-case alphabetic characters. Other characters are used verbatim.

Object/property type	Naming convention	Examples
Contexts	Aaaa or AaaaAaaaAaaa	Staff, GrandStaff
Layout Objects	Aaaa or AaaaAaaaAaaa	Slur, NoteHead
Engravers	Aaaa-aaa-engraver	Clef-engraver, Note-heads-engraver
Interfaces	aaa-aaa-interface	grob-interface, break-aligned-interface
Context Properties	aaa or aaaAaaaAaaa	alignAboveContext, skipBars
Layout Object Properties	aaa or aaa-aaa-aaa	direction, beam-thickness

As we shall see shortly, the properties of different types of object are modified by different commands, so it is useful to be able to recognize the types of objects and properties from their names.

4.1.4 Tweaking methods

The `\override` command

We have already met the commands `\set` and `\with`, used to change the properties of **contexts** and to remove and add **engravers**, in [\[Modifying context properties\]](#), [\[Adding and removing engravers\]](#). We must now introduce some more important commands.

The command to change the properties of **layout objects** is `\override`. Because this command has to modify internal properties deep within LilyPond its syntax is not as simple as the commands you have used so far. It needs to know precisely which property of which object in which context has to be modified, and what its new value is to be. Let’s see how this is done.

The general syntax of this command is:

```
\override Context.LayoutObject.layout-property = #value
```

This will set the property with the name *layout-property* of the layout object with the name *LayoutObject*, which is a member of the *Context* context, to the value *value*.

The *Context* may be omitted (and usually is) when the required context is unambiguously implied and is one of lowest level contexts, i.e., **Voice**, **ChordNames** or **Lyrics**, and we shall omit it in many of the following examples. We shall see later when it must be specified.

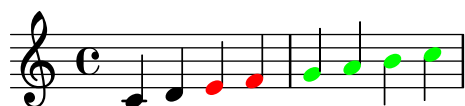
Later sections deal comprehensively with properties and their values, see [Secció 4.2.3 \[Types of properties\]](#), [pàgina 99](#). But in this section we shall use just a few simple properties and values which are easily understood in order to illustrate the format and use of these commands.

LilyPond’s primary expressions are musical items like notes, durations, and markups. More basic expressions like numbers, strings, and lists are processed in ‘Scheme mode’, which is invoked

by prefixing the value with ‘#’. Although the values may sometimes have a valid representation in LilyPond’s musical mode, this manual will always use ‘#’ for their entry for the sake of consistency. For more information about Scheme mode, see [Secció “LilyPond Scheme syntax” in *Extendre*](#).

`\override` is the most common command used in tweaking, and most of the rest of this chapter will be directed to presenting examples of how it is used. Here is a simple example to change the color of the note head:

```
c4 d
\override NoteHead.color = #red
e4 f |
\override NoteHead.color = #green
g4 a b c |
```



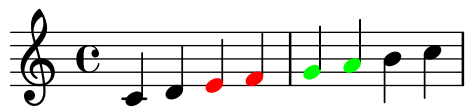
The `\revert` command

Once overridden, the property retains its new value until it is overridden again or a `\revert` command is encountered. The `\revert` command has the following syntax and causes the value of the property to revert to its original default value; note, not its previous value if several `\override` commands have been issued.

```
\revert Context.LayoutObject.layout-property
```

Again, just like *Context* in the `\override` command, *Context* is often not needed. It will be omitted in many of the following examples. Here we revert the color of the note head to the default value for the final two notes:

```
c4 d
\override NoteHead.color = #red
e4 f |
\override NoteHead.color = #green
g4 a
\revert NoteHead.color
b4 c |
```

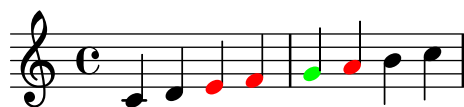


The `\once` prefix

Both the `\override` and the `\set` commands may be prefixed by `\once`. This causes the following `\override` or `\set` command to be effective only during the current musical moment before the property reverts back to its previous value (this can be different from the default if another `\override` is still in effect). Using the same example, we can change the color of a single note like this:

```
c4 d
\override NoteHead.color = #red
e4 f |
\once \override NoteHead.color = #green
g4 a
\revert NoteHead.color
b4 c |
```

b c |



The `\once` prefix may also be used in front of many predefined commands to limit their effect to one musical moment:

```
c4 d
\once \stemDown
e4 f |
g4 a
\once \hideNotes
b c |
```



However, predefined commands of the form `\...Neutral`, `\...Off` and `\un...` use `\revert` internally rather than `\override` so prefixing these with `\once` has no effect.

The `\overrideProperty` command

There is another form of the override command, `\overrideProperty`, which is occasionally required. We mention it here for completeness, but for details see [Secció “Difficult tweaks” in *Extendre*](#).

The `\tweak` command

The final tweaking command which is available is `\tweak`. This should be used when several objects occur at the same musical moment, but you only want to change the properties of selected ones, such as a single note within a chord. Using `\override` would affect all the notes within a chord, whereas `\tweak` affects just the following item in the input stream.

Here’s an example. Suppose we wish to change the size of the middle note head (the E) in a C major chord. Let’s first see what `\once \override` would do:

```
<c e g>4
\once \override NoteHead.font-size = #-3
<c e g>4
<c e g>4
```



We see the override affects *all* the note heads in the chord. This is because all the notes of a chord occur at the same *musical moment*, and the action of `\once` is to apply the override to all layout objects of the type specified which occur at the same musical moment as the `\override` command itself.

The `\tweak` command operates in a different way. It acts on the immediately following item in the input stream. In its simplest form, it is effective only on objects which are created directly from the following item, essentially note heads and articulations.

So to return to our example, the size of the middle note of a chord would be changed in this way:


```
<c e g>4
<c \tweak font-size #-3 e g>4
```



Note that the syntax of `\tweak` is different from that of the `\override` command. The context should not be specified; in fact, it would generate an error to do so. Both context and layout object are implied by the following item in the input stream. Note also that an equals sign should not be present. So the simple form of the `\tweak` command is

```
\tweak layout-property #value
```

A `\tweak` command can also be used to modify just one in a series of articulations, as shown here:

```
a4~"Black"
-\tweak color #red ~"Red"
-\tweak color #green _"Green"
```



Note that the `\tweak` command must be preceded by an articulation mark since the tweaked expression needs to be applied as an articulation itself. In case of multiple direction overrides (`~` or `_`), the leftmost override wins since it is applied last.

Objects such as stems and accidentals are created later, and not directly from the following event. It is still possible to use `\tweak` on such indirectly created objects by explicitly naming the layout object, provided that LilyPond can trace its origin back to the original event:

```
<\tweak Accidental.color #red cis4
\tweak Accidental.color #green es
g>
```



This long form of the `\tweak` command can be described as

```
\tweak LayoutObject.layout-property #value
```

The `\tweak` command must also be used to change the appearance of one of a set of nested tuplets which begin at the same musical moment. In the following example, the long tuplet bracket and the first of the three short brackets begin at the same musical moment, so any `\override` command would apply to both of them. In the example, `\tweak` is used to distinguish between them. The first `\tweak` command specifies that the long tuplet bracket is to be placed above the notes and the second one specifies that the tuplet number is to be printed in red on the first short tuplet bracket.

```
\tweak direction #up
\tuplet 3/4 {
  \tweak color #red
  \tuplet 3/2 { c8[ c c] }
```

```

\tuplet 3/2 { c8[ c c] }
\tuplet 3/2 { c8[ c c] }
}

```



If nested triplets do not begin at the same moment, their appearance may be modified in the usual way with `\override` commands:

```

\tuplet 3/2 { c8[ c c] }
\once \override TupletNumber.text = #tuplet-number::calc-fraction-text
\tuplet 3/2 {
  c8[ c]
  c8[ c]
  \once \override TupletNumber.transparent = ##t
  \tuplet 3/2 { c8[ c c] }
  \tuplet 3/2 { c8[ c c] }
}

```



Vegeu també

Notation Reference: [Secció “The tweak command”](#) in *Referència de la notació*.

The `\single` prefix

Suppose we wanted to emphasize particular note heads by coloring them red and increasing their size, and to make it easy suppose also we have defined a function to do this:

```

emphNoteHead = {
  \override NoteHead.color = #red
  \override NoteHead.font-size = 2
}
\relative c' {
  c4 a \once \emphNoteHead f d |
}

```



The `\once` prefix works fine to emphasize single notes or complete chords, but it cannot be used to emphasize a single note *within* a chord. Earlier we have seen how `\tweak` can be used to do this, see [\[The `\tweak` command\]](#), [pàgina 91](#). But `\tweak` cannot be used with a function; that's where `\single` comes in:

```

emphNoteHead = {
  \override NoteHead.color = #red
  \override NoteHead.font-size = 2
}

```

```
\relative c'' {
  <c a \single \emphNoteHead f d>4
}
```



In summary, `\single` converts overrides into tweaks so when there are several objects at the same point in musical time (like noteheads in a chord), `\single` will only affect a single one, the one generated by the immediately following music expression, in contrast to `\once` which will affect all of those objects.

By using `\single` in this way any shorthand function containing just overrides may be applied to individual notes in a chord. However, `\single` does not convert `\revert`, `\set` or `\unset` into tweaks.

Vegeu també

Learning Manual: [\[The `\tweak` command\]](#), pàgina 91, Secció 4.7.2 [\[Using variables for layout adjustments\]](#), pàgina 141.

4.2 The Internals Reference manual

4.2.1 Properties of layout objects

Suppose you have a slur in a score which, to your mind, appears too thin and you'd like to draw it a little heavier. How do you go about doing this? You know from the statements earlier about the flexibility of LilyPond that such a thing should be possible, and you would probably guess that an `\override` command would be needed. But is there a heaviness property for a slur, and if there is, how might it be modified? This is where the Internals Reference manual comes in. It contains all the information you might need to construct this and all other `\override` commands.

Before we look at the Internals Reference a word of warning. This is a **reference** document, which means there is little or no explanation contained within it: its purpose is to present information precisely and concisely. This means it might look daunting at first sight. Don't worry! The guidance and explanation presented here will enable you to extract the information from the Internals Reference for yourself with just a little practice.

Let's use a concrete example with a simple fragment of real music:

```
{
  \key es \major
  \time 6/8
  {
    r4 bes8 bes[( g]) g |
    g8[( es]) es d[( f]) as |
    as8 g
  }
  \addlyrics {
    The man who | feels love's sweet e -- | mo -- tion
  }
}
```



Suppose now that we decide we would like the slurs to be a little heavier. Is this possible? The slur is certainly a layout object, so the question is, ‘Is there a property belonging to a slur which controls the heaviness?’ To answer this we must look in the Internals Reference, or IR for short.

The IR for the version of LilyPond you are using may be found on the LilyPond website at <http://lilypond.org>. Go to the documentation page and click on the Internals Reference link. For learning purposes you should use the standard HTML version, not the ‘one big page’ or the PDF. For the next few paragraphs to make sense you will need to actually do this as you read.

Under the heading **Top** you will see five links. Select the link to the *Backend*, which is where information about layout objects is to be found. There, under the heading **Backend**, select the link to *All layout objects*. The page that appears lists all the layout objects used in your version of LilyPond, in alphabetic order. Select the link to Slur, and the properties of Slurs are listed.

An alternative way of finding this page is from the Notation Reference. On one of the pages that deals with slurs you may find a link to the Internals Reference. This link will take you directly to this page, but if you have an idea about the name of the layout object to be tweaked, it is easier to go straight to the IR and search there.

This Slur page in the IR tells us first that Slur objects are created by the `Slur_engraver`. Then it lists the standard settings. Browse through them looking for a property that might control the heaviness of slurs, and you should find

```
thickness (number)
1.2
Line thickness, generally measured in line-thickness
```

This looks a good bet to change the heaviness. It tells us that the value of `thickness` is a simple *number*, that the default value is 1.2, and that the units are in another property called `line-thickness`.

As we said earlier, there are few to no explanations in the IR, but we already have enough information to try changing the slur thickness. We see that the name of the layout object is `Slur`, that the name of the property to change is `thickness` and that the new value should be a number somewhat larger than 1.2 if we are to make slurs thicker.

We can now construct the `\override` command by simply substituting the values we have found for the names, omitting the context. Let’s use a very large value for the thickness at first, so we can be sure the command is working. We get:

```
\override Slur.thickness = #5.0
```

Don’t forget the `#` preceding the new value!

The final question is, ‘Where should this command be placed?’ While you are unsure and learning, the best answer is, ‘Within the music, before the first slur and close to it.’ Let’s do that:

```
{
  \key es \major
  \time 6/8
  {
    % Increase thickness of all following slurs from 1.2 to 5.0
    \override Slur.thickness = #5.0
    r4 bes8 bes[( g)] g |
    g8[( es)] es d[( f)] as |
    as8 g
  }
}
```

```

\addlyrics {
  The man who | feels love's sweet e -- | mo -- tion
}

```



and we see that the slur is indeed heavier.

So this is the basic way of constructing `\override` commands. There are a few more complications that we shall meet in later sections, but you now know all the essentials required to make up your own – but you will still need some practice. This is provided in the examples which follow.

Finding the context

But first, what if we had needed to specify the Context? What should it be? We could guess that slurs are in the Voice context, as they are clearly closely associated with individual lines of music, but can we be sure? To find out, go back to the top of the IR page describing the Slur, where it says ‘Slur objects are created by: Slur engraver’. So slurs will be created in whichever context the `Slur_engraver` is in. Follow the link to the `Slur_engraver` page. At the very bottom it tells us that `Slur_engraver` is part of seven Voice contexts, including the standard voice context, `Voice`, so our guess was correct. And because `Voice` is one of the lowest level contexts which is implied unambiguously by the fact that we are entering notes, we can omit it in this location.

Overriding once only

As you can see, *all* the slurs are thicker in the final example above. But what if we wanted just the first slur to be thicker? This is achieved with the `\once` command. Placed immediately before the `\override` command it causes it to change only the slur which begins on the **immediately following** note. If the immediately following note does not begin a slur the command has no effect at all – it is not remembered until a slur is encountered, it is simply discarded. So the command with `\once` must be repositioned as follows:

```

{
  \key es \major
  \time 6/8
  {
    r4 bes8
    % Increase thickness of immediately following slur only
    \once \override Slur.thickness = #5.0
    bes8[( g)] g |
    g8[( es)] es d[( f)] as |
    as8 g
  }
  \addlyrics {
    The man who | feels love's sweet e -- | mo -- tion
  }
}

```



Now only the first slur is made heavier.

The `\once` command can also be used before the `\set` command.

Reverting

Finally, what if we wanted just the first two slurs to be heavier? Well, we could use two commands, each preceded by `\once` placed immediately before each of the notes where the slurs begin:

```
{
  \key es \major
  \time 6/8
  {
    r4 bes8
    % Increase thickness of immediately following slur only
    \once \override Slur.thickness = #5.0
    bes[( g)] g |
    % Increase thickness of immediately following slur only
    \once \override Slur.thickness = #5.0
    g8[( es)] es d[( f)] as |
    as8 g
  }
  \addlyrics {
    The man who | feels love's sweet e -- | mo -- tion
  }
}
```



or we could omit the `\once` command and use the `\revert` command to return the `thickness` property to its default value after the second slur:

```
{
  \key es \major
  \time 6/8
  {
    r4 bes8
    % Increase thickness of all following slurs from 1.2 to 5.0
    \override Slur.thickness = #5.0
    bes[( g)] g |
    g8[( es)] es
    % Revert thickness of all following slurs to default of 1.2
    \revert Slur.thickness
    d8[( f)] as |
    as8 g
  }
  \addlyrics {
    The man who | feels love's sweet e -- | mo -- tion
  }
}
```

}



The `\revert` command can be used to return any property changed with `\override` back to its default value. You may use whichever method best suits what you want to do.

That concludes our introduction to the IR, and the basic method of tweaking. Several examples follow in the later sections of this Chapter, partly to introduce you to some of the additional features of the IR, and partly to give you more practice in extracting information from it. These examples will contain progressively fewer words of guidance and explanation.

4.2.2 Properties found in interfaces

Suppose now that we wish to print the lyrics in italics. What form of `\override` command do we need to do this? We first look in the IR page listing ‘All layout objects’, as before, and look for an object that might control lyrics. We find `LyricText`, which looks right. Clicking on this shows the settable properties for lyric text. These include the `font-series` and `font-size`, but nothing that might give an italic shape. This is because the shape property is one that is common to all font objects, so, rather than including it in every layout object, it is grouped together with other similar common properties and placed in an **Interface**, the `font-interface`.

So now we need to learn how to find the properties of interfaces, and to discover what objects use these interface properties.

Look again at the IR page which describes `LyricText`. At the bottom of the page is a list of clickable interfaces which `LyricText` supports. The list has several items, including `font-interface`. Clicking on this brings up the properties associated with this interface, which are also properties of all the objects which support it, including `LyricText`.

Now we see all the user-settable properties which control fonts, including `font-shape(symbol)`, where `symbol` can be set to `upright`, `italics` or `caps`.

You will notice that `font-series` and `font-size` are also listed there. This immediately raises the question: Why are the common font properties `font-series` and `font-size` listed under `LyricText` as well as under the interface `font-interface` but `font-shape` is not? The answer is that `font-series` and `font-size` are changed from their global default values when a `LyricText` object is created, but `font-shape` is not. The entries in `LyricText` then tell you the values for those two properties which apply to `LyricText`. Other objects which support `font-interface` will set these properties differently when they are created.

Let’s see if we can now construct the `\override` command to change the lyrics to italics. The object is `LyricText`, the property is `font-shape` and the value is `italic`. As before, we’ll omit the context.

As an aside, although it is an important one, note that some properties take values that are symbols, like `italic`, and must be preceded by an apostrophe, `'`. Symbols are then read internally by LilyPond. Note the distinction from arbitrary text strings, which would appear as `"a text string"`; for more details about symbols and strings, see [Secció “Scheme tutorial” in *Extendre*](#).

So we see that the `\override` command needed to print the lyrics in italics is:

```
\override LyricText.font-shape = #'italic
```

This should be placed just in front of the lyrics we wish to affect, like so:

```

{
  \key es \major
  \time 6/8
  {
    r4 bes8 bes[( g]) g |
    g8[( es]) es d[( f]) as |
    as8 g
  }
  \addlyrics {
    \override LyricText.font-shape = #'italic
    The man who | feels love's sweet e -- | mo -- tion
  }
}

```



and the lyrics are all printed in italics.

Nota: In lyrics always leave whitespace between the final syllable and the terminating brace.

Vegeu també

Extending: [Secció “Scheme tutorial” in *Extendre*](#).

4.2.3 Types of properties

So far we have seen two types of property: **number** and **symbol**. To be valid, the value given to a property must be of the correct type and obey the rules for that type. The type of property is always shown in brackets after the property name in the IR. Here is a list of the types you may need, together with the rules for that type, and some examples. You must always add a hash symbol, #, of course, to the front of these values when they are entered in the `\override` command, even if the value itself already starts with #. We only give examples for constants here: if you want to compute a value using Scheme, see [Secció “Calculations in Scheme” in *Extendre*](#).

Property type	Rules	Examples
Boolean	Either True or False, represented by #t or #f	#t, #f
Dimension (in staff space)	A decimal number (in units of staff space)	2.5, 0.34
Direction	A valid direction constant or its numerical equivalent (0 or CENTER indicate a neutral direction)	LEFT, CENTER, UP, 1, -1
Integer	A whole number	3, -1
List	A sequence of constants or symbols separated by spaces, enclosed in parentheses and preceded by an apostrophe (quote mark)	'(left-edge staff-bar), '(1), '(), '(1.0 0.25 0.5)
Markup	Any valid markup	\markup { \italic "cresc." }, "bagpipe"

Moment	A fraction of a whole note constructed with the make-moment function	<code>(ly:make-moment 1/4)</code> , <code>(ly:make-moment 3/8)</code>
Number	Any positive or negative, possibly decimal, value	<code>3</code> , <code>-2.45</code>
Pair (of numbers)	Two numbers separated by a ‘space . space’ and enclosed in brackets preceded by an apostrophe	<code>'(2 . 3.5)</code> , <code>'(0.1 . -3.2)</code>
Symbol	Any of the set of permitted symbols for that property, preceded by an apostrophe	<code>'italic</code> , <code>'inside</code>
Unknown	A procedure, or <code>#f</code> to cause no action	<code>bend::print</code> , <code>ly:text-interface::print</code> , <code>#f</code>
Vector	Constants enclosed in <code>#(...)</code> .	<code>##t ##t #f</code>

Vegeu també

Extending: [Secció “Scheme tutorial” in *Extendre*](#).

4.3 Appearance of objects

Let us now put what we have learned into practice with a few examples which show how tweaks may be used to change the appearance of the printed music.

4.3.1 Visibility and color of objects

In the educational use of music we might wish to print a score with certain elements omitted as an exercise for the student, who is required to supply them. As a simple example, let us suppose the exercise is to supply the missing bar lines in a piece of music. But the bar lines are normally inserted automatically. How do we prevent them printing?

Before we tackle this, let us remember that object properties are grouped in what are called *interfaces* – see [Secció 4.2.2 \[Properties found in interfaces\]](#), [pàgina 98](#). This is simply to group together those properties that may be used together to tweak a graphical object – if one of them is allowed for an object, so are the others. Some objects then use the properties in some interfaces, others use them from other interfaces. The interfaces which contain the properties used by a particular grob are listed in the IR at the bottom of the page describing that grob, and those properties may be viewed by looking at those interfaces.

We explained how to find information about grobs in [Secció 4.2.1 \[Properties of layout objects\]](#), [pàgina 94](#). Using the same approach, we go to the IR to find the layout object which prints bar lines. Going via *Backend* and *All layout objects* we find there is a layout object called `BarLine`. Its properties include two that control its visibility: `break-visibility` and `stencil`. `BarLine` also supports a number of interfaces, including the `grob-interface`, where we find the `transparent` and the `color` properties. All of these can affect the visibility of bar lines (and, of course, by extension, many other layout objects too.) Let’s consider each of these in turn.

The stencil property

This property controls the appearance of the bar lines by specifying the symbol (glyph) which should be printed. In common with many other properties, it can be set to print nothing by setting its value to `#f`. Let’s try it, as before, omitting the implied Context, Voice:

```
{
  \time 12/16
  \override BarLine.stencil = ##f
  c4 b8 c d16 c d8 |
```

```

g,8 a16 b8 c d4 e16 |
e8
}

```



The bar lines are still printed. What is wrong? Go back to the IR and look again at the page giving the properties of BarLine. At the top of the page it says “Barline objects are created by: Bar_engraver”. Go to the Bar_engraver page. At the bottom it gives a list of Contexts in which the bar engraver operates. All of them are of the type **Staff**, so the reason the `\override` command failed to work as expected is because **Barline** is not in the default **Voice** context. If the context is specified incorrectly, the command simply does not work. No error message is produced, and nothing is logged in the log file. Let’s try correcting it by adding the correct context:

```

{
  \time 12/16
  \override Staff.BarLine.stencil = ##f
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}

```



Now the bar lines have vanished. Setting the **stencil** property to **#f** is such a frequent operation that there is a shorthand for it called `\omit`:

```

{
  \time 12/16
  \omit Staff.BarLine
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}

```



Note, though, that setting the **stencil** property to **#f** will cause errors when the dimensions of the object are required for correct processing. For example, errors will be generated if the **stencil** property of the **NoteHead** object is set to **#f**. If this is the case, you can instead use the **point-stencil** function, which sets the stencil to an object with zero size:

```

{
  c4 c
  \once \override NoteHead.stencil = #point-stencil
  c4 c
}

```



The break-visibility property

We see from the `BarLine` properties in the IR that the `break-visibility` property requires a vector of three booleans. These control respectively whether bar lines are printed at the end of a line, in the middle of lines, and at the beginning of lines. For our example we want all bar lines to be suppressed, so the value we need is `##(#f #f #f)` (also available under the name `all-invisible`). Let's try that, remembering to include the `Staff` context. Note also that in writing this value we have `##` before the opening parenthesis. One `#` is required as part of vector constant syntax, and the first `#` is required, as always, to precede the value itself in the `\override` command.

```
{
  \time 12/16
  \override Staff.BarLine.break-visibility = ##(#f #f #f)
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



And we see this too removes all the bar lines.

The transparent property

We see from the properties specified in the `grob-interface` page in the IR that the `transparent` property is a boolean. This should be set to `#t` to make the grob transparent. In this next example let us make the time signature invisible rather than the bar lines. To do this we need to find the grob name for the time signature. Back to the 'All layout objects' page in the IR to find the properties of the `TimeSignature` layout object. This is produced by the `Time_signature_engraver` which you can check also lives in the `Staff` context and also supports the `grob-interface`. So the command to make the time signature transparent is:

```
{
  \time 12/16
  \override Staff.TimeSignature.transparent = ##t
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



Again, setting the `transparent` property is a rather frequent operation, so we have a shorthand for it called `\hide`:

```
{
  \time 12/16
  \hide Staff.TimeSignature
  c4 b8 c d16 c d8 |
}
```

```
g,8 a16 b8 c d4 e16 |
e8
}
```



In either case, the time signature is gone, but this command leaves a gap where the time signature should be. Maybe this is what is wanted for an exercise for the student to fill it in, but in other circumstances a gap might be undesirable. To remove it, the stencil for the time signature should be set to `#f` instead:

```
{
  \time 12/16
  \omit Staff.TimeSignature
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



and the difference is obvious: setting the stencil to `#f` (possibly via `\omit`) removes the object entirely; making the object `transparent` (which can be done using `\hide`) leaves it where it is, but makes it invisible.

The color property

Finally let us try making the bar lines invisible by coloring them white. (There is a difficulty with this in that the white bar line may or may not blank out the staff lines where they cross. You may see in some of the examples below that this happens unpredictably. The details of why this is so and how to control it are covered in [Secció “Painting objects white” in Referència de la notació](#). But at the moment we are learning about color, so please just accept this limitation for now.)

The `grob-interface` specifies that the color property value is a list, but there is no explanation of what that list should be. The list it requires is actually a list of values in internal units, but, to avoid having to know what these are, several ways are provided to specify colors. The first way is to use one of the ‘normal’ colors listed in the first table in [Secció “List of colors” in Referència de la notació](#). To set the bar lines to white we write:

```
{
  \time 12/16
  \override Staff.BarLine.color = #white
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



and again, we see the bar lines are not visible. Note that *white* is not preceded by an apostrophe – it is not a symbol, but a *variable*. When evaluated, it provides the list of internal values required to set the color to white. The other colors in the normal list are variables too. To convince yourself this is working you might like to change the color to one of the other variables in the list.

The second way of changing the color is to use the list of X11 color names in the second list in *Secció “List of colors” in Referència de la notació*. However, these are mapped to the actual values by the function `x11-color` which converts X11 color symbols into the list of internal values like this:

```
{
  \time 12/16
  \override Staff.BarLine.color = #(x11-color 'white)
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



Note that in this case the function `x11-color` takes a symbol as an argument, so the symbol must be preceded by an apostrophe to keep it from being evaluated as a variable, and the whole function call has to be enclosed in parentheses.

There is another function, one which converts RGB values into internal colors – the `rgb-color` function. This takes three arguments giving the intensities of the red, green and blue colors. These take values in the range 0 to 1. So to set the color to red the value should be `(rgb-color 1 0 0)` and to white it should be `(rgb-color 1 1 1)`:

```
{
  \time 12/16
  \override Staff.BarLine.color = #(rgb-color 1 1 1)
  c4 b8 c d16 c d8 |
  g,8 a16 b8 c d4 e16 |
  e8
}
```



Finally, there is also a grey scale available as part of the X11 set of colors. These range from black, `'grey0`, to white, `'grey100`, in steps of 1. Let's illustrate this by setting all the layout objects in our example to various shades of grey:

```
{
  \time 12/16
  \override Staff.StaffSymbol.color = #(x11-color 'grey30)
  \override Staff.TimeSignature.color = #(x11-color 'grey60)
  \override Staff.Clef.color = #(x11-color 'grey60)
  \override Voice.NoteHead.color = #(x11-color 'grey85)
  \override Voice.Stem.color = #(x11-color 'grey85)
  \override Staff.BarLine.color = #(x11-color 'grey10)
```

```

c4 b8 c d16 c d8 |
g,8 a16 b8 c d4 e16 |
e8
}

```



Note the contexts associated with each of the layout objects. It is important to get these right, or the commands will not work! Remember, the context is the one in which the appropriate engraver is placed. The default context for engravers can be found by starting from the layout object, going from there to the engraver which produces it, and on the engraver page in the IR it tells you in which context the engraver will normally be found.

4.3.2 Size of objects

Let us begin by looking again at the earlier example (see [\[Nesting music expressions\]](#), [pàgina \[Nesting music expressions\]](#)) which showed how to introduce a new temporary staff, as in an *Secció “ossia”* in *Glossari musical*.

```

\new Staff ="main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
    <<
    { f8 c c }
    \new Staff \with {
      alignAboveContext = #"main" }
    { f8 f c }
    >>
    r4 |
  }
}

```



Ossia are normally written without clef and time signature, and are usually printed slightly smaller than the main staff. We already know now how to remove the clef and time signature – we simply set the stencil of each to `#f`, as follows:

```

\new Staff ="main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
    <<
    { f8 c c }
    \new Staff \with {
      alignAboveContext = #"main"

```

```

    }
    {
        \omit Staff.Clef
        \omit Staff.TimeSignature
        { f8 f c }
    }
>>
r4 |
}
}

```



where the extra pair of braces after the `\with` clause are required to ensure the enclosed overrides and music are applied to the ossia staff.

But what is the difference between modifying the staff context by using `\with` and modifying the stencils of the clef and the time signature with `\override`, or in this case `\omit`? The main difference is that changes made in a `\with` clause are made at the time the context is created, and remain in force as the **default** values for the duration of that context, whereas `\set` or `\override` commands embedded in the music are dynamic – they make changes synchronized with a particular point in the music. If changes are unset or reverted using `\unset` or `\revert` they return to their default values, which will be the ones set in the `\with` clause, or if none have been set there, the normal default values.

Some context properties can be modified only in `\with` clauses. These are those properties which cannot sensibly be changed after the context has been created. `alignAboveContext` and its partner, `alignBelowContext`, are two such properties – once the staff has been created its alignment is decided and it would make no sense to try to change it later.

The default values of layout object properties can also be set in `\with` clauses. Simply use the normal `\override` command leaving out the context name, since this is unambiguously defined as the context which the `\with` clause is modifying. If fact, an error will be generated if a context is specified in this location.

So we could replace the example above with

```

\new Staff ="main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
  }
  { f8 c c }
  \new Staff \with {
    alignAboveContext = #"main"
    % Don't print clefs in this staff
    \override Clef.stencil = ##f
    % Don't print time signatures in this staff
    \override TimeSignature.stencil = ##f
  }
  { f8 f c }
}

```

```
>>
r4 |
}
}
```



It turns out that we can also employ the shorthands `\hide` and `\omit` for setting the `transparent` property and clearing the `stencil` here, leading to the result

```
\new Staff ="main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
    <<
    { f8 c c }
    \new Staff \with {
      alignAboveContext = #"main"
      % Don't print clefs in this staff
      \omit Clef
      % Don't print time signatures in this staff
      \omit TimeSignature
    }
    { f8 f c }
  }
  >>
  r4 |
}
}
```



Finally we come to changing the size of layout objects.

Some layout objects are created as glyphs selected from a typeface font. These include note heads, accidentals, markup, clefs, time signatures, dynamics and lyrics. Their size is changed by modifying the `font-size` property, as we shall shortly see. Other layout objects such as slurs and ties – in general, spanner objects – are drawn individually, so there is no `font-size` associated with them. These objects generally derive their size from the objects to which they are attached, so usually there is no need to change their size manually. Still other properties such as the length of stems and bar lines, thickness of beams and other lines, and the separation of staff lines all need to be modified in special ways.

Returning to the ossia example, let us first change the font-size. We can do this in two ways. We can either change the size of the fonts of each object type, like `NoteHeads` with commands like


```
\override NoteHead.font-size = #-2
```

or we can change the size of all fonts by setting a special property, `fontSize`, using `\set`, or by including it in a `\with` clause (but without the `\set`).

```
\set fontSize = #-2
```

Both of these statements would cause the font size to be reduced by 2 steps from its previous value, where each step reduces or increases the size by approximately 12%.

Let's try it in our ossia example:

```
\new Staff = "main" {
  \relative g' {
    r4 g8 g c4 c8 d |
    e4 r8
  }
  <<
  { f8 c c }
  \new Staff \with {
    alignAboveContext = #"main"
    \omit Clef
    \omit TimeSignature
    % Reduce all font sizes by ~24%
    fontSize = #-2
  }
  { f8 f c }
  >>
  r4 |
}
}
```



This is still not quite right. The note heads and flags are smaller, but the stems are too long in proportion and the staff lines are spaced too widely apart. These need to be scaled down in proportion to the font reduction. The next sub-section discusses how this is done.

4.3.3 Length and thickness of objects

Distances and lengths in LilyPond are generally measured in staff-spaces, the distance between adjacent lines in the staff, (or occasionally half staff spaces) while most **thickness** properties are measured in units of an internal property called **line-thickness**. For example, by default, the lines of hairpins are given a thickness of 1 unit of **line-thickness**, while the **thickness** of a note stem is 1.3. Note, though, that some thickness properties are different; for example, the thickness of beams is controlled by the value of the **beam-thickness** property, which is measured in staff-spaces.

So how are lengths to be scaled in proportion to the font size? This can be done with the help of a special function called **magstep** provided for exactly this purpose. It takes one argument, the change in font size (`#-2` in the example above) and returns a scaling factor suitable for reducing other objects in proportion. It is used like this:

```
\new Staff = "main" {
```

```

\relative g' {
  r4 g8 g c4 c8 d |
  e4 r8
  <<
    { f8 c c }
    \new Staff \with {
      alignAboveContext = #"main"
      \omit Clef
      \omit TimeSignature
      fontSize = #-2
      % Reduce stem length and line spacing to match
      \override StaffSymbol.staff-space = #(magstep -2)
    }
    { f8 f c }
  >>
  r4 |
}

```



Since the length of stems and many other length-related properties are always calculated relative to the value of the `staff-space` property these are automatically scaled down in length too. Note that this affects only the vertical scale of the ossia – the horizontal scale is determined by the layout of the main music in order to remain synchronized with it, so it is not affected by any of these changes in size. Of course, if the scale of all the main music were changed in this way then the horizontal spacing would be affected. This is discussed later in the layout section.

This, then, completes the creation of an ossia. The sizes and lengths of all other objects may be modified in analogous ways.

For small changes in scale, as in the example above, the thickness of the various drawn lines such as bar lines, beams, hairpins, slurs, etc does not usually require global adjustment. If the thickness of any particular layout object needs to be adjusted this can be best achieved by overriding its `thickness` property. An example of changing the thickness of slurs was shown above in [Secció 4.2.1 \[Properties of layout objects\], pàgina 94](#). The thickness of all drawn objects (i.e., those not produced from a font) may be changed in the same way.

4.4 Placement of objects

4.4.1 Automatic behavior

There are some objects in musical notation that belong to the staff and there are other objects that should be placed outside the staff. These are called within-staff objects and outside-staff objects respectively.

Within-staff objects are those that are located on the staff – note heads, stems, accidentals, etc. The positions of these are usually fixed by the music itself – they are vertically positioned on specific lines of the staff or are tied to other objects that are so positioned. Collisions of note heads, stems and accidentals in closely set chords are normally avoided automatically. There are commands and overrides which can modify this automatic behavior, as we shall shortly see.

Objects belonging outside the staff include things such as rehearsal marks, text and dynamic markings. LilyPond's rule for the vertical placement of outside-staff objects is to place them as close to the staff as possible but not so close that they collide with any other object. LilyPond uses the `outside-staff-priority` property to determine the order in which the objects should be placed, as follows.

First, LilyPond places all the within-staff objects. Then it sorts the outside-staff objects according to their `outside-staff-priority`. The outside-staff objects are taken one by one, beginning with the object with the lowest `outside-staff-priority`, and placed so that they do not collide with any objects that have already been placed. That is, if two outside-staff grobs are competing for the same space, the one with the lower `outside-staff-priority` will be placed closer to the staff. If two objects have the same `outside-staff-priority` the one encountered first will be placed closer to the staff.

In the following example all the markup texts have the same priority (since it is not explicitly set). Note that 'Text3' is automatically positioned close to the staff again, nestling under 'Text2'.

```
c2~"Text1"
c2~"Text2" |
c2~"Text3"
c2~"Text4" |
```



Staves are also positioned, by default, as closely together as possible (subject to a minimum separation). If notes project a long way towards an adjacent staff they will force the staves further apart only if an overlap of the notation would otherwise occur. The following example demonstrates this 'nestling' of the notes on adjacent staves:

```
<<
\new Staff {
  \relative c' { c4 a, }
}
\new Staff {
  \relative c'''' { c4 a, }
}
>>
```



4.4.2 Within-staff objects

We have already seen how the commands `\voiceXXX` affect the direction of slurs, ties, fingering and everything else which depends on the direction of the stems – see [\[Explicitly instantiating voices\]](#), [pàgina](#) [\[Explicitly instantiating voices\]](#). These commands are essential when writing polyphonic music to permit interweaving melodic lines to be distinguished. But occasionally it may be

necessary to override this automatic behavior. This can be done for whole sections of music or even for an individual note. The property which controls this behavior is the **direction** property of each layout object. We first explain what this does, and then introduce a number of ready-made commands which avoid your having to code explicit overrides for the more common modifications.

Some layout objects like slurs and ties curve, bend or point either up or down; others like stems and flags also move to right or left when they point up or down. This is controlled automatically when **direction** is set.

The direction property

The following example shows in bar 1 the default behavior of stems, with those on high notes pointing down and those on low notes pointing up, followed by four notes with all stems forced down, four notes with all stems forced up, and finally four notes reverted back to the default behavior.

```
a4 g c a |
\override Stem.direction = #DOWN
a4 g c a |
\override Stem.direction = #UP
a4 g c a |
\revert Stem.direction
a4 g c a |
```



Here we use the constants **DOWN** and **UP**. These have the values **-1** and **+1** respectively, and these numerical values may be used instead. The value **0** may also be used in some cases. It is simply treated as meaning **UP** for stems, but for some objects it means ‘center’. There is a constant, **CENTER** which has the value **0**.

However, these explicit overrides are not usually used, as there are simpler equivalent pre-defined commands available. Here is a table of the commonest. The meaning of each is stated where it is not obvious.

Down/Left	Up/Right	Revert	Effect
<code>\arpeggioArrowDown</code>	<code>\arpeggioArrowUp</code>	<code>\arpeggioNormal</code>	Arrow is at bottom, at top, or no arrow
<code>\dotsDown</code>	<code>\dotsUp</code>	<code>\dotsNeutral</code>	Direction of movement to avoid staff lines
<code>\dynamicDown</code>	<code>\dynamicUp</code>	<code>\dynamicNeutral</code>	
<code>\phrasingSlurDown</code>	<code>\phrasingSlurUp</code>	<code>\phrasingSlurNeutral</code>	Note: distinct from slur commands
<code>\slurDown</code>	<code>\slurUp</code>	<code>\slurNeutral</code>	
<code>\stemDown</code>	<code>\stemUp</code>	<code>\stemNeutral</code>	
<code>\textSpannerDown</code>	<code>\textSpannerUp</code>	<code>\textSpannerNeutral</code>	Text entered as spanner is below/above staff
<code>\tieDown</code>	<code>\tieUp</code>	<code>\tieNeutral</code>	
<code>\tupletDown</code>	<code>\tupletUp</code>	<code>\tupletNeutral</code>	Tuplets are below/above notes

The neutral/normal variants of these commands are implemented using `\revert` and may **not** be preceded by `\once`. If you wish to limit the effect of the other commands (which are

implemented using `\override`) to a single timestep, you can precede them with `\once` like you would do with explicit overrides.

Fingering

The placement of fingering on single notes can also be controlled by the `direction` property, but changing `direction` has no effect on chords. As we shall see, there are special commands which allow the fingering of individual notes of chords to be controlled, with the fingering being placed above, below, to the left or to the right of each note.

First, here's the effect of `direction` on the fingering attached to single notes. The first bar shows the default behaviour, and the following two bars shows the effect of specifying `DOWN` and `UP`:

```
c4-5 a-3 f-1 c'-5 |
\override Fingering.direction = #DOWN
c4-5 a-3 f-1 c'-5 |
\override Fingering.direction = #UP
c4-5 a-3 f-1 c'-5 |
```



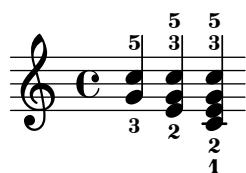
However, overriding the `direction` property is not the easiest way of manually setting the fingering above or below the notes; using `_` or `^` instead of `-` before the fingering number is usually preferable. Here is the previous example using this method:

```
c4-5 a-3 f-1 c'-5 |
c4_5 a_3 f_1 c'_5 |
c4^5 a^3 f^1 c'^5 |
```



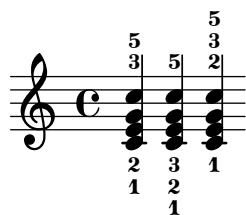
The `direction` property is ignored for chords, but the directional prefixes, `_` and `^` do work. By default, the fingering is automatically placed both above and below the notes of a chord, as shown:

```
<c-5 g-3>4
<c-5 g-3 e-2>4
<c-5 g-3 e-2 c-1>4
```



but this may be overridden to manually force all or any of the individual fingering numbers above or below:

```
<c-5 g-3 e-2 c-1>4
<c^5 g_3 e_2 c_1>4
<c^5 g^3 e^2 c_1>4
```



Even greater control over the placement of fingering of the individual notes in a chord is possible by using the `\set fingeringOrientations` command. The format of this command is:

```
\set fingeringOrientations = #'([up] [left/right] [down])
```

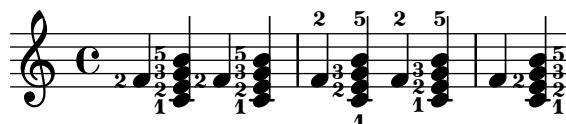
`\set` is used because `fingeringOrientations` is a property of the `Voice` context, created and used by the `New_fingering_engraver`.

The property may be set to a list of one to three values. It controls whether fingerings may be placed above (if `up` appears in the list), below (if `down` appears), to the left (if `left` appears), or to the right (if `right` appears). Conversely, if a location is not listed, no fingering is placed there. LilyPond takes these constraints and works out the best placement for the fingering of the notes of the following chords. Note that `left` and `right` are mutually exclusive – fingering may be placed only on one side or the other, not both.

Nota: To control the placement of the fingering of a single note using this command it is necessary to write it as a single note chord by placing angle brackets round it.

Here are a few examples:

```
\set fingeringOrientations = #'(left)
<f-2>4
<c-1 e-2 g-3 b-5>4
\set fingeringOrientations = #'(left)
<f-2>4
<c-1 e-2 g-3 b-5>4 |
\set fingeringOrientations = #'(up left down)
<f-2>4
<c-1 e-2 g-3 b-5>4
\set fingeringOrientations = #'(up left)
<f-2>4
<c-1 e-2 g-3 b-5>4 |
\set fingeringOrientations = #'(right)
<f-2>4
<c-1 e-2 g-3 b-5>4
```



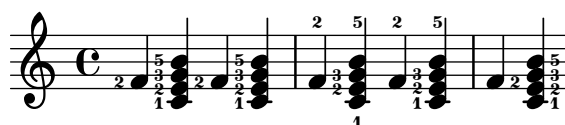
If the fingering seems a little crowded the `font-size` could be reduced. The default value can be seen from the `Fingering` object in the IR to be `-5`, so let's try `-7`:

```
\override Fingering.font-size = #-7
\set fingeringOrientations = #'(left)
<f-2>4
<c-1 e-2 g-3 b-5>4
\set fingeringOrientations = #'(left)
<f-2>4
<c-1 e-2 g-3 b-5>4 |
```

```

\set fingeringOrientations = #'(up left down)
<f-2>4
<c-1 e-2 g-3 b-5>4
\set fingeringOrientations = #'(up left)
<f-2>4
<c-1 e-2 g-3 b-5>4 |
\set fingeringOrientations = #'(right)
<f-2>4
<c-1 e-2 g-3 b-5>4

```



4.4.3 Outside-staff objects

Outside-staff objects are automatically placed to avoid collisions. There are several ways to override the automatic placement if the positioning is not optimum.

The outside-staff-priority property

Objects with the lower value of the `outside-staff-priority` property are placed nearer to the staff, and other outside-staff objects are then raised as far as necessary to avoid collisions. The `outside-staff-priority` is defined in the `grob-interface` and so is a property of all layout objects. By default it is set to `#f` for all within-staff objects, and to a numerical value appropriate to each outside-staff object when the object is created. The following table shows the default numerical values for some of the commonest outside-staff objects.

Note the unusual names for some of the objects: spanner objects are automatically created to control the vertical positioning of grobs which (might) start and end at different musical moments, so changing the `outside-staff-priority` of the underlying grob will have no effect. For example, changing `outside-staff-priority` of the `Hairpin` object will have no effect on the vertical positioning of hairpins – you must change `outside-staff-priority` of the associated `DynamicLineSpanner` object instead. This override must be placed at the start of the spanner, which might include several linked hairpins and dynamics.

Layout Object	Priority	Controls position of:
RehearsalMark	1500	Rehearsal marks
MetronomeMark	1000	Metronome marks
VoltaBracketSpanner	600	Volta brackets
TextScript	450	Markup text
MultiMeasureRestText	450	Markup text over full-bar rests
OttavaBracket	400	Ottava brackets
TextSpanner	350	Text spanners
DynamicLineSpanner	250	All dynamic markings
BarNumber	100	Bar numbers
TrillSpanner	50	Spanning trills

Here is an example showing the default placement of some of these.

```

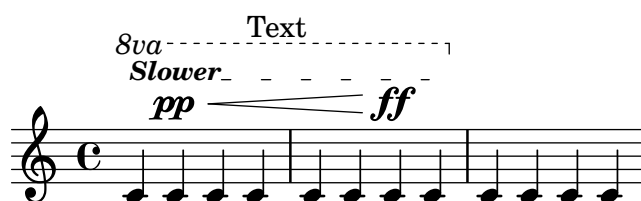
% Set details for later Text Spanner
\override TextSpanner.bound-details.left.text
  = \markup { \small \bold Slower }
% Place dynamics above staff

```

```

\dynamicUp
% Start Ottava Bracket
\ottava #1
c'4 \startTextSpan
% Add Dynamic Text and hairpin
c4\pp\<
c4
% Add Text Script
c4~Text |
c4 c
% Add Dynamic Text and terminate hairpin
c4\ff c \stopTextSpan |
% Stop Ottava Bracket
\ottava #0
c,4 c c c |

```



This example also shows how to create Text Spanners – text with extender lines above a section of music. The spanner extends from the `\startTextSpan` command to the `\stopTextSpan` command, and the format of the text is defined by the `\override TextSpanner` command. For more details see [Secció “Text spanners” in Referència de la notació](#).

It also shows how ottava brackets are created.

If the default values of `outside-staff-priority` do not give you the placing you want, the priority of any of the objects may be overridden. Suppose we would like the ottava bracket to be placed below the text spanner in the example above. All we need to do is to look up the priority of `OttavaBracket` in the IR or in the tables above, and reduce it to a value lower than that of a `TextSpanner`, remembering that `OttavaBracket` is created in the `Staff` context:

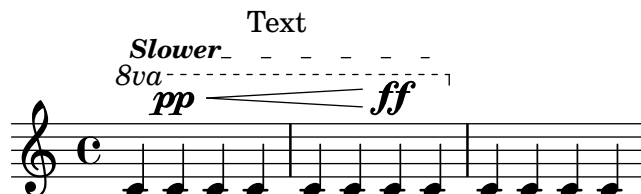
```

% Set details for later Text Spanner
\override TextSpanner.bound-details.left.text
  = \markup { \small \bold Slower }
% Place dynamics above staff
\dynamicUp
% Place following Ottava Bracket below Text Spanners
\once \override Staff.OttavaBracket.outside-staff-priority = #340
% Start Ottava Bracket
\ottava #1
c'4 \startTextSpan
% Add Dynamic Text
c4\pp
% Add Dynamic Line Spanner
c4\<
% Add Text Script
c4~Text |
c4 c
% Add Dynamic Text
c4\ff c \stopTextSpan |

```



```
% Stop Ottava Bracket
\ottava #0
c,4 c c c |
```



Note that some of these objects, in particular bar numbers, metronome marks and rehearsal marks, live by default in the `Score` context, so be sure to use the correct context when these are being overridden.

Slurs by default are classed as within-staff objects, but they often appear above the staff if the notes to which they are attached are high on the staff. This can push outside-staff objects such as articulations too high, as the slur will be placed first. The `avoid-slur` property of the articulation can be set to `'inside` to bring the articulation inside the slur, but the `avoid-slur` property is effective only if the `outside-staff-priority` is also set to `#f`. Alternatively, the `outside-staff-priority` of the slur can be set to a numerical value to cause it to be placed along with other outside-staff objects according to that value. Here's an example showing the effect of the two methods:

```
c4( c^\markup { \tiny \sharp } d4.) c8 |
c4(
\once \override TextScript.avoid-slur = #'inside
\once \override TextScript.outside-staff-priority = ##f
c4^\markup { \tiny \sharp } d4.) c8 |
\once \override Slur.outside-staff-priority = #500
c4( c^\markup { \tiny \sharp } d4.) c8 |
```



Changing the `outside-staff-priority` can also be used to control the vertical placement of individual objects, although the results may not always be desirable. Suppose we would like “Text3” to be placed above “Text4” in the example under Automatic behavior, above (see [Secció 4.4.1 \[Automatic behavior\]](#), [pàgina 109](#)). All we need to do is to look up the priority of `TextScript` in the IR or in the tables above, and increase the priority of “Text3” to a higher value:

```
c2^"Text1"
c2^"Text2" |
\once \override TextScript.outside-staff-priority = #500
c2^"Text3"
c2^"Text4" |
```



This certainly lifts “Text3” above “Text4” but it also lifts it above “Text2”, and “Text4” now drops down. Perhaps this is not so good. What we would really like to do is to position all the annotation at the same distance above the staff. To do this, we clearly will need to space the notes out horizontally to make more room for the text. This is done using the `\textLengthOn` command.

The `\textLengthOn` command

By default, text produced by markup takes up no horizontal space as far as laying out the music is concerned. The `\textLengthOn` command reverses this behavior, causing the notes to be spaced out as far as is necessary to accommodate the text:

```
\textLengthOn % Cause notes to space out to accommodate text
c2^"Text1"
c2^"Text2" |
c2^"Text3"
c2^"Text4" |
```



The command to revert to the default behavior is `\textLengthOff`. Alternatively, `\once` may be used with `\textLengthOn` if the effect is to be limited to just a single musical moment. The corresponding spacing behavior for rehearsal marks and tempo indications is independently controlled with the commands `\markLengthOn` and `\markLengthOff`.

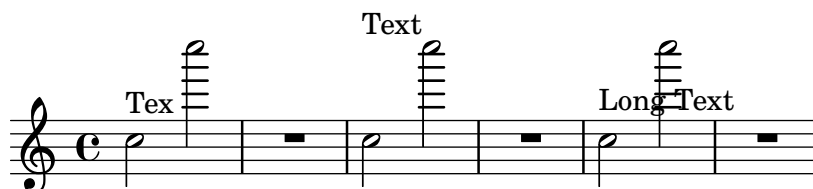
Markup text will also avoid notes which project above the staff. If this is not desired, the automatic displacement upwards may be turned off by setting the priority to `#f`. Here’s an example to show how markup text interacts with such notes.

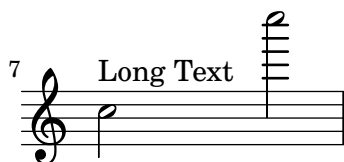
```
% This markup is short enough to fit without collision
c2^"Tex" c'' |
R1 |

% This is too long to fit, so it is displaced upwards
c,,2^"Text" c'' |
R1 |

% Turn off collision avoidance
\once \override TextScript.outside-staff-priority = ##f
c,,2^"Long Text" c'' |
R1 |

% Turn off collision avoidance
\once \override TextScript.outside-staff-priority = ##f
\textLengthOn % and turn on textLengthOn
c,,2^"Long Text" % Spaces at end are honored
c''2 |
```





Dynamics placement

Dynamic markings will normally be positioned beneath the staff, but may be positioned above with the `\dynamicUp` command. They will be positioned vertically relative to the note to which they are attached, and will float below (or above) all within-staff objects such as phrasing slurs and bar numbers. This can give quite acceptable results, as this example shows:

```
\clef "bass"
\key aes \major
\time 9/8
\dynamicUp
bes4.~\f\< \(\ bes4 bes8 des4\ff\> c16 bes\! |
ees,2.~\)\mf ees4 r8 |
```



However, if the notes and attached dynamics are close together the automatic placement will avoid collisions by displacing later dynamic markings further away, but this may not be the optimum placement, as this rather artificial example shows:

```
\dynamicUp
a4\f b\mf a\mp b\p
```



Should a similar situation arise in ‘real’ music, it may be preferable to space out the notes a little further, so the dynamic markings can all fit at the same vertical distance from the staff. We were able to do this for markup text by using the `\textLengthOn` command, but there is no equivalent command for dynamic marks. So we shall have to work out how to do this using `\override` commands.

Grob sizing

First we must learn how grobs are sized. All grobs have a reference point defined within them which is used to position them relative to their parent object. This point in the grob is then positioned at a horizontal distance, `X-offset`, and at a vertical distance, `Y-offset`, from its parent. The horizontal extent of the object is given by a pair of numbers, `X-extent`, which say where the left and right edges are relative to the reference point. The vertical extent is similarly defined by a pair of numbers, `Y-extent`. These are properties of all grobs which support the `grob-interface`.

By default, outside-staff objects are given a width of zero so that they may overlap in the horizontal direction. This is done by the trick of making the leftmost extent infinity and the rightmost extent minus infinity by setting the `extra-spacing-width` to `'(+inf.0 . -inf.0)`. To ensure they do not overlap in the horizontal direction we must override this value of `extra-spacing-width` to give them a little extra spacing. The units are the space between two staff lines, so moving the left edge half a unit to the left and the right edge half a unit to the right should do it:

```
\override DynamicText.extra-spacing-width = #'(-0.5 . 0.5)
```

Let's see if this works in our previous example:

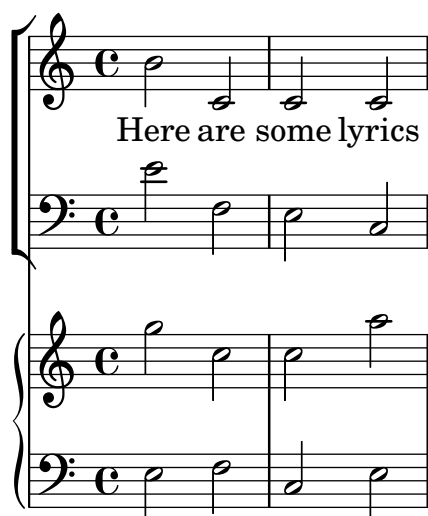
```
\dynamicUp
% Extend width by 1 staff space
\override DynamicText.extra-spacing-width = #'(-0.5 . 0.5)
a4\f b\mf a\mp b\p
```



This looks better, but maybe we would prefer the dynamic marks to be aligned along the same baseline rather than going up and down with the notes. The property to do this is `staff-padding` which is covered in the section on collisions (see [Secció 4.6 \[Collisions of objects\]](#), [pàgina 123](#)).

4.5 Vertical spacing

As a rule, LilyPond's vertical spacing of musical objects is pretty good. Let's see how it does with a simple song, with 2 voices and piano accompaniment:



There's nothing wrong with the default vertical spacing. However, let's assume that you're working with a publisher with some specific requirements for vertical spacing of staves and lyrics: they want the lyrics spaced away from any notes, they want the piano accompaniment spaced away from the vocal line and they want the two piano staves pushed together tightly. Let's start with the lyrics.

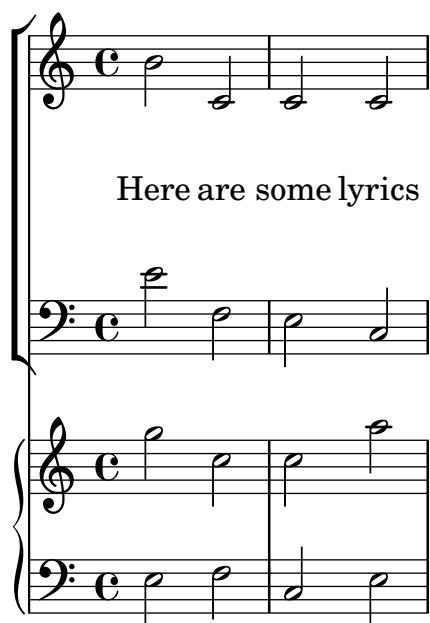
Lyrics sit within a system, and therefore the commands to space them are found in [Secció "Flexible vertical spacing within systems"](#) in *Referència de la notació*. It tells us that lyrics are `non-staff` lines and therefore the command to change their spacing will refer to the `nonstaff` property. Spacing them away from the staff to which they relate (the top line) will use the `relatedstaff` property. Spacing them from the lower line will use the `unrelatedstaff` property. The vocal parts are part of a `VerticalAxisGroup`, so we need to adjust its properties. Let's try it and see if it works.

```
<<
\new ChoirStaff
<<
```

```

\new Staff {
  \new Voice = "music" {
    b'2 c' c' c'
  }
}
\new Lyrics \with {
  \override VerticalAxisGroup.
    nonstaff-relatedstaff-spacing.padding = #5
  \override VerticalAxisGroup.
    nonstaff-unrelatedstaff-spacing.padding = #5
}
\lyricsto "music" {
  Here are some lyrics
}
\new Staff {
  \clef bass e'2 f e c
}
>>
\new PianoStaff
<<
  \new Staff {
    g''2 c'' c'' a''
  }
  \new Staff {
    \clef bass e2 f c e
  }
>>
>>

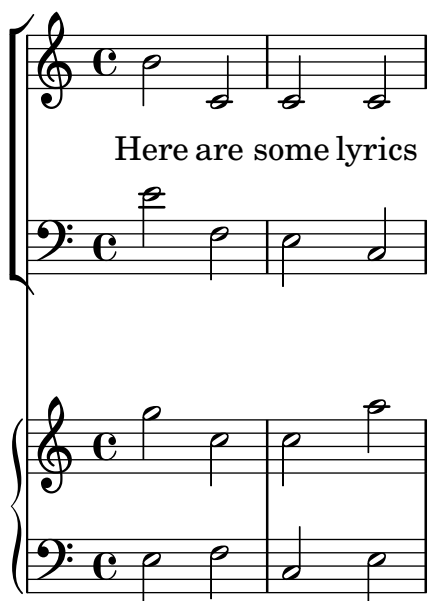
```



Well - yes it does, but perhaps too well. When we set the `padding` to 5, LilyPond adds 5 staff spaces to the distance between objects, which is too much for us here. We'll use 2.

Next, let's move the piano music away from the vocal parts. The vocal music is a **ChoirStaff**, so we need to increase the spacing between that group of staves and the piano staff below. We'll do this by changing the **basic-distance** of the **StaffGrouper**'s **staffgroup-staff-spacing**.

```
<<
  \new ChoirStaff \with {
    \override StaffGrouper.
      staffgroup-staff-spacing.basic-distance = #15
  }
<<
  \new Staff {
    \new Voice = "music" {
      b'2 c' c' c'
    }
  }
  \new Lyrics \with {
    \override VerticalAxisGroup.
      nonstaff-relatedstaff-spacing.padding = #2
    \override VerticalAxisGroup.
      nonstaff-unrelatedstaff-spacing.padding = #2
  }
  \lyricsto "music" {
    Here are some lyrics
  }
  \new Staff {
    \clef bass e'2 f e c
  }
>>
\new PianoStaff
<<
  \new Staff {
    g''2 c'' c'' a''
  }
  \new Staff {
    \clef bass e2 f c e
  }
>>
>>
```



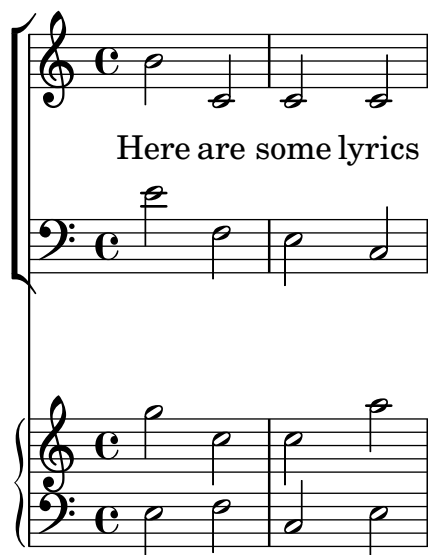
Excellent. Now just for the last requirement to make the piano staves closer together. To do this, we again alter the properties of the `StaffGrouper`, but this time we're going to reduce both the `basic-distance` and the `padding`. We can do this as shown below.

```
<<
  \new ChoirStaff \with {
    \override StaffGrouper.
      staffgroup-staff-spacing.basic-distance = #15
  }
<<
  \new Staff {
    \new Voice = "music" {
      b'2 c' c' c'
    }
  }
  \new Lyrics \with {
    \override VerticalAxisGroup.
      nonstaff-relatedstaff-spacing.padding = #2
    \override VerticalAxisGroup.
      nonstaff-unrelatedstaff-spacing.padding = #2
  }
  \lyricsto "music" {
    Here are some lyrics
  }
  \new Staff {
    \clef bass e'2 f e c
  }
>>
\new PianoStaff \with {
  \override StaffGrouper.staff-staff-spacing = #'(
    (basic-distance . 0)
    (padding . 0))
}
<<
  \new Staff {
    g''2 c'' c'' a''
  }
```

```

\new Staff {
  \clef bass e2 f c e
}
>>
>>

```



That’s put them really close together – but it’s what the publisher wanted. They could be moved further apart by altering the `padding` or `basic-distance` if wanted.

There are many ways of altering vertical spacing. A key point to remember is that the spacing between objects in a `StaffGroup` (like `GrandStaff` or `PianoStaff` groups) is controlled by the spacing variables of the `StaffGrouper`. Spacing from ungrouped staves (like `Lyrics` and `Staff`) is controlled by the variables of the `VerticalAxisGroup`. See the [Secció “Flexible vertical spacing paper variables”](#) in *Referència de la notació* and [Secció “Flexible vertical spacing within systems”](#) in *Referència de la notació* for more details.

4.6 Collisions of objects

4.6.1 Moving objects

This may come as a surprise, but LilyPond is not perfect. Some notation elements can overlap. This is unfortunate, but in fact rather rare. Usually the need to move objects is for clarity or aesthetic reasons – they would look better with a little more or a little less space around them.

There are three main approaches to resolving overlapping notation. They should be considered in the following order:

1. The **direction** of one of the overlapping objects may be changed using the predefined commands listed above for within-staff objects (see [Secció 4.4.2 \[Within-staff objects\]](#), [pàgina 110](#)). Stems, slurs, beams, ties, dynamics, text and tuplets may be repositioned easily in this way. The limitation is that you have a choice of only two positions, and neither may be suitable.
2. The **object properties**, which LilyPond uses when positioning layout objects, may be modified using `\override`. The advantages of making changes to this type of property are (a) that some other objects will be moved automatically if necessary to make room and (b) the single override can apply to all instances of the same type of object. Such properties include:

- **direction**

This has already been covered in some detail – see [Secció 4.4.2 \[Within-staff objects\]](#), [pàgina 110](#).

- **padding, right-padding, staff-padding**

As an object is being positioned the value of its **padding** property specifies the gap that must be left between itself and the nearest edge of the object against which it is being positioned. Note that it is the **padding** value of the object **being placed** that is used; the **padding** value of the object which is already placed is ignored. Gaps specified by **padding** can be applied to all objects which support the **side-position-interface**.

Instead of **padding**, the placement of groups of accidentals is controlled by **right-padding**. This property is to be found in the **AccidentalPlacement** object which, note, lives in the **Staff** context. In the typesetting process the note heads are typeset first and then the accidentals, if any, are added to the left of the note heads using the **right-padding** property to determine the separation from the note heads and between individual accidentals. So only the **right-padding** property of the **AccidentalPlacement** object has any effect on the placement of the accidentals.

The **staff-padding** property is closely related to the **padding** property: **padding** controls the minimum amount of space between any object which supports the **side-position-interface** and the nearest other object (generally the note or the staff lines); **staff-padding** applies only to those objects which are always set outside the staff – it controls the minimum distance from the staff to the outside-staff object. Note that **staff-padding** has no effect on objects that are positioned relative to the note rather than the staff, even though it may be overridden without error for such objects – it is simply ignored.

To discover which padding property is required for the object you wish to reposition, you need to return to the IR and look up the object’s properties. Be aware that the padding properties might not be located in the obvious object, so look in objects that appear to be related.

All padding values are measured in staff spaces. For most objects, this value is set by default to be around 1.0 or less (it varies with each object). It may be overridden if a larger (or smaller) gap is required.

- **self-alignment-X**

This property can be used to align the object to the left, to the right, or to center it with respect to the parent object’s reference point. It may be used with all objects which support the **self-alignment-interface**. In general these are objects that contain text. The values are **LEFT**, **RIGHT** or **CENTER**. Alternatively, a numerical value between **-1** and **+1** may be specified, where **-1** is left-aligned, **+1** is right-aligned, and numbers in between move the text progressively from left-aligned to right-aligned. Numerical values greater than **1** may be specified to move the text even further to the left, or less than **-1** to move the text even further to the right. A change of **1** in the value corresponds to a movement of half the text’s length.

- **extra-spacing-width**

This property is available for all objects which support the **item-interface**. It takes two numbers, the first is added to the leftmost extent and the second is added to the rightmost extent. Negative numbers move the edge to the left, positive to the right, so to widen an object the first number must be negative, the second positive. Note that not all objects honor both numbers. For example, the **Accidental** object only takes notice of the first (left edge) number.

- **staff-position**

staff-position is a property of the **staff-symbol-referencer-interface**, which is supported by objects which are positioned relative to the staff. It specifies the vertical position of the object relative to the center line of the staff in half staff-spaces. It is useful in resolving collisions between layout objects like multi-measure rests, ties and notes in different voices.

- **horizontal-shift**

Within a voice, all the notes occurring at the same musical moment are grouped into a note column, and a **NoteColumn** object is created to control the horizontal positioning of that group of notes (see “Note columns” in [\[Explicitly instantiating voices\]](#), [pàgina \[undefined\]](#)). If *and only if* two or more note columns within a single Staff context, both with stems in the same direction, occur at the same musical moment, the values of their **horizontal-shift** properties are used to rank them and the columns in the higher ranks are progressively offset to avoid collisions of the noteheads. This property is set by the **\voiceXXX** commands and may be overridden directly with an **\override** command or, more usually, by the **\shiftOn** commands. Note that this property is used to *rank* the note columns for off-setting - it does not specify the magnitude of the offset, which is progressively increased in steps based on the note head’s width for each rank. The steps are usually of half a note head’s width, but may be a full note head’s width when a closely spaced group of notes is involved.

- **force-hshift**

The **force-hshift** property is a property of a **NoteColumn** (actually of the **note-column-interface**). Changing it permits a note column to be moved in situations where the note columns overlap. Note that it has no effect on note columns that do not overlap. It is specified in units appropriate to a note column, viz. the note head width of the first voice note. It should be used in complex situations where the normal **\shiftOn** commands (see [\[Explicitly instantiating voices\]](#), [pàgina \[undefined\]](#)) do not resolve the note conflict satisfactorily. It is preferable to the **extra-offset** property for this purpose as there is no need to work out the distance in staff-spaces, and moving the notes into or out of a **NoteColumn** affects other actions such as merging note heads.

3. Finally, when all else fails, objects may be manually repositioned relative to the staff center line vertically, or by displacing them by any distance to a new position. The disadvantages are that the correct values for the repositioning have to be worked out, often by trial and error, for every object individually, and, because the movement is done after LilyPond has placed all other objects, the user is responsible for avoiding any collisions that might ensue. But the main difficulty with this approach is that the repositioning values may need to be reworked if the music is later modified. The properties that can be used for this type of manual repositioning are:

extra-offset

This property applies to any layout object supporting the **grob-interface**. It takes a pair of numbers which specify the extra displacement in the horizontal and vertical directions. Negative numbers move the object to the left or down. The units are staff-spaces. The extra displacement is made after the typesetting of objects is finished, so an object may be repositioned anywhere without affecting anything else.

positions

This is most useful for manually adjusting the slope and height of beams, slurs, and tuplets. It takes a pair of numbers giving the position of the left and right ends of the beam, slur, etc. relative to the center line of the staff. Units are staff-spaces. Note, though, that slurs and phrasing slurs cannot be repositioned

by arbitrarily large amounts. LilyPond first generates a list of possible positions for the slur and by default finds the slur that “looks best”. If the `positions` property has been overridden the slur that is closest to the requested positions is selected from the list.

A particular object may not have all of these properties. It is necessary to go to the IR to look up which properties are available for the object in question.

Here is a list of the objects which are most likely to be involved in collisions, together with the name of the object which should be looked up in the IR in order to discover which properties should be used to move them.

Object type	Object name
Articulations	Script
Beams	Beam
Dynamics (vertically)	DynamicLineSpanner
Dynamics (horizontally)	DynamicText
Fingerings	Fingering
Rehearsal / Text marks	RehearsalMark
Slurs	Slur
Text e.g. <code>~"text"</code>	TextScript
Ties	Tie
Tuplets	TupletBracket

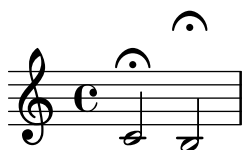
4.6.2 Fixing overlapping notation

Let’s now see how the properties in the previous section can help to resolve overlapping notation.

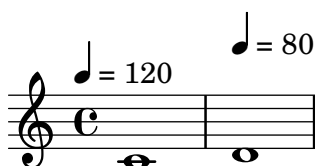
The padding property

The `padding` property can be set to increase (or decrease) the distance between symbols that are printed above or below notes.

```
c2\fermata
\override Script.padding = #3
b2\fermata
```



```
% This will not work, see below
\override MetronomeMark.padding = #3
\tempo 4 = 120
c1 |
% This works
\override Score.MetronomeMark.padding = #3
\tempo 4 = 80
d1 |
```



Note in the second example how important it is to figure out what context handles a certain object. Since the `MetronomeMark` object is handled in the `Score` context, property changes in the `Voice` context will not be noticed. For more details, see [Secció “Modifying properties” in Referència de la notació](#).

If the `padding` property of an object is increased when that object is in a stack of objects being positioned according to their `outside-staff-priority`, then that object and all objects outside it are moved.

The right-padding property

The `right-padding` property affects the spacing between the accidental and the note to which it applies. It is not often required, but the default spacing may be wrong for certain special accidental glyphs or combination of glyphs used in some microtonal music. These have to be entered by overriding the accidental stencil with a markup containing the desired symbol(s), like this:

```
sesquisharp = \markup { \sesquisharp }
\relative c'' {
  c4
  % This prints a sesquisharp but the spacing is too small
  \once \override Accidental.stencil = #ly:text-interface::print
  \once \override Accidental.text = #sesquisharp
  cis4 c
  % This improves the spacing
  \once \override Score.AccidentalPlacement.right-padding = #0.6
  \once \override Accidental.stencil = #ly:text-interface::print
  \once \override Accidental.text = #sesquisharp
  cis4 |
}
```



This necessarily uses an override for the accidental stencil which will not be covered until later. The stencil type must be a procedure, here changed to print the contents of the `text` property of `Accidental`, which itself is set to be a sesquisharp sign. This sign is then moved further away from the note head by overriding `right-padding`.

The staff-padding property

`staff-padding` can be used to align objects such as dynamics along a baseline at a fixed distance from the staff, when no other notation forces them further from the staff. It is not a property of `DynamicText` but of `DynamicLineSpanner`. This is because the baseline should apply equally to **all** dynamics, including those created as extended spanners. So this is the way to align the dynamic marks in the example taken from the previous section:

```
\override DynamicLineSpanner.staff-padding = #3
a4\f b\mf a\p b\mp
```



The self-alignment-X property

The following example shows how to adjust the position of a string fingering object relative to a note's stem by aligning the right edge with the reference point of the parent note:

```
\voiceOne
<a\2>
\once \override StringNumber.self-alignment-X = #RIGHT
<a\2>
```



The staff-position property

Multimeasure rests in one voice can collide with notes in another. Since these rests are typeset centered between the bar lines, it would require significant effort for LilyPond to figure out which other notes might collide with it, since all the current collision handling between notes and between notes and rests is done only for notes and rests that occur at the same time. Here's an example of a collision of this type:

```
<< { c4 c c c } \\ { R1 } >>
```



The best solution here is to move the multimeasure rest down, since the rest is in voice two. The default in `\voiceTwo` (i.e. in the second voice of a `<<{...} \\ {...}>>` construct) is that `staff-position` is set to -4 for `MultiMeasureRest`, so we need to move it, say, four half-staff spaces down to -8.

```
<<
  { c4 c c c }
  \\
  \override MultiMeasureRest.staff-position = #-8
  { R1 }
>>
```



This is better than using, for example, `extra-offset`, because the ledger line above the rest is inserted automatically.

The extra-offset property

The `extra-offset` property provides complete control over the positioning of an object both horizontally and vertically.

In the following example, the second fingering is moved a little to the left, and 1.8 staff space downwards:

```

\stemUp
f4-5
\once \override Fingering.extra-offset = #'(-0.3 . -1.8)
f4-5

```



The positions property

The `positions` property allows the vertical position and hence the slope of tuplets, slurs, phrasing slurs and beams to be controlled manually.

Here's an example in which the phrasing slur and slur collide:

```
a8 \(( a'16 ) a \)
```



One possibility would be to move the two ends of the phrasing slur higher. We can try setting the left end to 2.5 staff-spaces above the centre line and the right end to 4.5 above, and LilyPond will select the phrasing slur from the candidates it has found with its end points closest to these:

```

\once \override PhrasingSlur.positions = #'(2.5 . 4.5)
a8 \(( a'16 ) a \)

```



This is an improvement, but why not lower the right end of the slur a little? If you try it you'll find it can't be done in this way. That's because there are no candidate slurs lower than the one already selected, and in this case the `positions` property has no effect. However, ties, slurs and phrasing slurs *can* be positioned and shaped very precisely when necessary. To learn how to do this, see [Secció "Modifying ties and slurs" in Referència de la notació](#).

Here's a further example. We see that the beams collide with the ties:

```

{
  \time 4/2
  <<
    { c1~ 2. e8 f }
    \\\
    {
      e'8 e e e
      e e e e
      f2 g
    }
  >>
  <<
    { c,,1~ 2. e8 f }
    \\\
    {

```

```

      e'8 e e e
      e e e e
      f2 g
    }
  >>
}

```



This can be resolved by manually moving both ends of the beam up from their position at 1.81 staff-spaces below the center line to, say, 1:

```

{
  \time 4/2
  <<
    { c1~ 2. e8 f }
    \\
    {
      \override Beam.positions = #'(-1 . -1)
      e'8 e e e
      e e e e
      f2 g
    }
  >>
  <<
    { c,,1~ 2. e8 f }
    \\
    {
      e'8 e e e
      e e e e
      f2 g
      \revert Beam.positions
    }
  >>
}

```



Note that the override continues to apply in the second voice of the second measure of eighth notes, but not to any of the beams in the first voice, even those in the later second measure. As soon as the override should no longer apply it should be reverted, as shown.

The force-hshift property

We can now see how to apply the final corrections to the Chopin example introduced at the end of [\[I'm hearing Voices\]](#), [pàgina \[I'm hearing Voices\]](#), which was left looking like this:

```

\new Staff \relative c'' {
  \key aes \major
  <<
    { c2 aes4. bes8 }
  >>
}

```

```

\\
{ <ees, c>2 des }
\\
\\
{ aes'2 f4 fes }
>> |
<c ees aes c>1 |
}

```



The inner note of the first chord (i.e. the A-flat in the fourth Voice) need not be shifted away from the note column of the higher note, so we use `\shiftOff`.

In the second chord we prefer the F to line up with the A-flat and the lowest note to be positioned slightly right to avoid a collision of stems. We achieve this by setting `force-hshift` in the `NoteColumn` of the low D-flat to move it to the right by half a staff-space, and setting `force-hshift` for the F to zero. Note that we use `\once` to avoid the settings propagating beyond the immediate musical moment, although in this small example the `\once` and the second `\override` in Voice four could be omitted. This would not be good practice.

Here's the final result:

```

\new Staff \relative c'' {
  \key aes \major
  <<
    { c2 aes4. bes8 }
    \\
    { <ees, c>2 \once \override NoteColumn.force-hshift = 0.5 des }
    \\
    \\
    { \once \shiftOff aes'2 \once \shiftOff f4 fes }
  >> |
  <c ees aes c>1 |
}

```



4.6.3 Real music example

We end this section on Tweaks by showing the steps to be taken to deal with a tricky example which needs several tweaks to produce the desired output. The example has been deliberately chosen to illustrate the use of the Notation Reference to resolve unusual problems with notation. It is not representative of the more usual engraving process, so please do not let these difficulties put you off! Fortunately, difficulties like these are not very common!

The example is from Chopin's *Première Ballade*, Op. 23, bars 6 to 9, the transition from the opening *Lento* to *Moderato*. Here, first, is what we want the output to look like, but to avoid over-complicating the example too much we have left out the dynamics, fingering and pedalling.



We note first that the right hand part in the third bar requires four voices. These are the five beamed eighth notes, the tied C, the half-note D which is merged with the eighth note D, and the dotted quarter note F-sharp, which is also merged with the eighth note at the same pitch. Everything else is in a single voice, so the easiest way is to introduce these extra three voices temporarily at the time they are needed. If you have forgotten how to do this, look at [\[I'm hearing Voices\]](#), [pàgina \[undefined\]](#) and [\[Explicitly instantiating voices\]](#), [pàgina \[undefined\]](#). Here we choose to use explicitly instantiated voices for the polyphonic passage, as LilyPond is better able to avoid collisions if all voices are instantiated explicitly in this way.

So let us begin by entering the notes as two variables, setting up the staff structure in a score block, and seeing what LilyPond produces by default:

```
rhMusic = \relative c' {
  \new Voice {
    r2 c4. g8 |
    bes1~ |
    \time 6/4
    bes2. r8
    % Start polyphonic section of four voices
    <<
      { c,8 d fis bes a } % continuation of main voice
      \new Voice {
        \voiceTwo
        c,8~ 2
      }
      \new Voice {
        \voiceThree
        s8 d2
      }
      \new Voice {
        \voiceFour
        s4 fis4.
      }
    >> |
    g2. % continuation of main voice
  }
}

lhMusic = \relative c' {
  r2 <c g ees>2 |
  <d g, d>1 |
  r2. d,,4 r4 r |
  r4
}

\score {
```

```

\new PianoStaff <<
  \new Staff = "RH" <<
    \key g \minor
    \rhMusic
  >>
  \new Staff = "LH" <<
    \key g \minor
    \clef "bass"
    \lhMusic
  >>
>>
}

```



All the notes are right, but the appearance is far from satisfactory. The tie collides with the change in time signature, some notes are not merged together, and several notation elements are missing. Let's first deal with the easier things. We can easily add the left hand slur and the right hand phrasing slur, since these were all covered in the Tutorial. Doing this gives:

```

rhMusic = \relative c' {
  \new Voice {
    r2 c4.\( g8 |
    bes1~ |
    \time 6/4
    bes2. r8
    % Start polyphonic section of four voices
    <<
      { c,8 d fis bes a } % continuation of main voice
      \new Voice {
        \voiceTwo
        c,8~ 2
      }
      \new Voice {
        \voiceThree
        s8 d2
      }
      \new Voice {
        \voiceFour
        s4 fis4.
      }
    >> |
    g2.\) % continuation of main voice
  }
}

lhMusic = \relative c' {

```

```

r2 <c g ees>2( |
<d g, d>1) |
r2. d,,4 r4 r |
r4
}

\score {
  \new PianoStaff <<
    \new Staff = "RH" <<
      \key g \minor
      \rhMusic
    >>
    \new Staff = "LH" <<
      \key g \minor
      \clef "bass"
      \lhMusic
    >>
  >>
}

```



The first bar is now correct. The second bar contains an arpeggio and is terminated by a double bar line. How do we do these, as they have not been mentioned in this Learning Manual? This is where we need to turn to the Notation Reference. Looking up ‘arpeggio’ and ‘bar line’ in the index quickly shows us that an arpeggio is produced by appending `\arpeggio` to a chord, and a double bar line is produced by the `\bar "||"` command. That’s easily done. We next need to correct the collision of the tie with the time signature. This is best done by moving the tie upwards. Moving objects was covered earlier in [Secció 4.6.1 \[Moving objects\], pàgina 123](#), which says that objects positioned relative to the staff can be moved vertically by overriding their `staff-position` property, which is specified in half staff spaces relative to the center line of the staff. So the following override placed just before the first tied note would move the tie up to 3.5 half staff spaces above the center line:

```
\once \override Tie.staff-position = #3.5
```

This completes bar two, giving:

```

rhMusic = \relative c' {
  \new Voice {
    r2 c4.\( g8 |
    \once \override Tie.staff-position = #3.5
    bes1~ |
    \bar "||"
    \time 6/4
    bes2. r8
    % Start polyphonic section of four voices
  }
}

```

```

    { c,8 d fis bes a } % continuation of main voice
    \new Voice {
      \voiceTwo
      c,8~ 2
    }
    \new Voice {
      \voiceThree
      s8 d2
    }
    \new Voice {
      \voiceFour
      s4 fis4.
    }
  >> |
  g2.\) % continuation of main voice
}
}

lhMusic = \relative c' {
  r2 <c g ees>2( |
  <d g, d>1)\arpeggio |
  r2. d,,4 r4 r |
  r4
}

\score {
  \new PianoStaff <<
    \new Staff = "RH" <<
      \key g \minor
      \rhMusic
    >>
    \new Staff = "LH" <<
      \key g \minor
      \clef "bass"
      \lhMusic
    >>
  >>
}

```



On to bar three and the start of the Moderato section. The tutorial showed how to add a tempo indication with the `\tempo` command, so adding “Moderato” is easy. But how do we merge notes in different voices together? This is where we need to turn again to the Notation Reference for help. A search for “merge” in the Notation Reference index quickly leads us to the commands for merging differently headed and differently dotted notes in [Secció “Collision](#)

resolution” in *Referència de la notació*. In our example we need to merge both types of note for the duration of the polyphonic section in bar 3, so using the information we find in the Notation Reference we add

```
\mergeDifferentlyHeadedOn
\mergeDifferentlyDottedOn
```

to the start of that section and

```
\mergeDifferentlyHeadedOff
\mergeDifferentlyDottedOff
```

to the end, giving:



These overrides have merged the two F-sharp notes, but not the two on D. Why not? The answer is there in the same section in the Notation Reference – notes being merged must have stems in opposite directions and two notes cannot be merged successfully if there is a third note in the same note column. Here the two D’s both have upward stems and there is a third note – the C. We know how to change the stem direction using `\stemDown`, and the Notation Reference also says how to move the C – apply a shift using one of the `\shift` commands. But which one? The C is in voice two which has shift off, and the two D’s are in voices one and three, which have shift off and shift on, respectively. So we have to shift the C a further level still using `\shift0nn` to avoid it interfering with the two D’s. Applying these changes gives:

```
rhMusic = \relative c'' {
  \new Voice {
    r2 c4.\( g8 |
    \once \override Tie.staff-position = #3.5
    bes1~ |
    \bar "||"
    \time 6/4
    bes2.\tempo "Moderato" r8
    \mergeDifferentlyHeadedOn
    \mergeDifferentlyDottedOn
    % Start polyphonic section of four voices
    <<
    { c,8 d fis bes a } % continuation of main voice
    \new Voice {
      \voiceTwo
      % Move the c2 out of the main note column
      % so the merge will work
      c,8~ \shift0nn c2
    }
    \new Voice {
      \voiceThree
      % Stem on the d2 must be down to permit merging
      s8 \stemDown d2
    }
  }
}
```

```

    }
    \new Voice {
      \voiceFour
      s4 fis4.
    }
  >> |
  \mergeDifferentlyHeadedOff
  \mergeDifferentlyDottedOff
  g2.\) % continuation of main voice
}
}

lhMusic = \relative c' {
  r2 <c g ees>2( |
  <d g, d>1)\arpeggio |
  r2. d,,4 r4 r |
  r4
}

\score {
  \new PianoStaff <<
    \new Staff = "RH" <<
      \key g \minor
      \rhMusic
    >>
    \new Staff = "LH" <<
      \key g \minor
      \clef "bass"
      \lhMusic
    >>
  >>
}

```



Nearly there. Only two problems remain: The downward stem on the merged D should not be there, and the C would be better positioned to the right of the D's. We know how to do both of these from the earlier tweaks: we make the stem transparent, and move the C with the `force-hshift` property. Here's the final result:

```

rhMusic = \relative c'' {
  \new Voice {
    r2 c4.\( g8 |
    \once \override Tie.staff-position = #3.5
    bes1~ |
    \bar "||"
  }
}

```

```

\time 6/4
bes2.\tempo "Moderato" r8
\mergeDifferentlyHeadedOn
\mergeDifferentlyDottedOn
% Start polyphonic section of four voices
<<
  { c,8 d fis bes a } % continuation of main voice
  \new Voice {
    \voiceTwo
    c,8~
    % Reposition the c2 to the right of the merged note
    \once \override NoteColumn.force-hshift = #1.0
    % Move the c2 out of the main note column
    % so the merge will work
    \shiftOnn
    c2
  }
  \new Voice {
    \voiceThree
    s8
    % Stem on the d2 must be down to permit merging
    \stemDown
    % Stem on the d2 should be invisible
    \tweak Stem.transparent ##t
    d2
  }
  \new Voice {
    \voiceFour
    s4 fis4.
  }
  >> |
\mergeDifferentlyHeadedOff
\mergeDifferentlyDottedOff
g2.\) % continuation of main voice
}
}

lhMusic = \relative c' {
  r2 <c g ees>2( |
  <d g, d>1)\arpeggio |
  r2. d,,4 r4 r |
  r4
}

\score {
  \new PianoStaff <<
    \new Staff = "RH" <<
      \key g \minor
      \rhMusic
    >>
    \new Staff = "LH" <<
      \key g \minor

```

```

\clef "bass"
\lhMusic
>>
>>
}

```



4.7 Further tweaking

4.7.1 Other uses for tweaks

Tying notes across voices

The following example demonstrates how to connect notes in different voices using ties. Normally, only notes in the same voice can be connected with ties. By using two voices, with the tied notes in one of them



and removing the first up-stem and its flag in that voice, the tie appears to cross voices:

```

<<
{
  \once \omit Stem
  \once \omit Flag
  b8~ 8\noBeam
}
\\
{ b8[ g] }
>>

```



Vegeu també

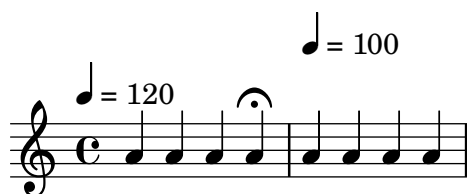
Learning Manual: [\[The \once prefix\]](#), pàgina 90, [\[The stencil property\]](#), pàgina 100.

Simulating a fermata in MIDI

For outside-staff objects it is usually better to override the object's `stencil` property rather than its `transparent` property when you wish to remove it from the printed output. Setting the `stencil` property of an object to `#f` will remove that object entirely from the printed output. This means it has no effect on the placement of other objects placed relative to it.

For example, if we wished to change the metronome setting in order to simulate a fermata in the MIDI output we would not want the metronome markings to appear in the printed output, and we would not want it to influence the spacing between the two systems or the positions of adjacent annotations on the staff. So setting its `stencil` property to `#f` would be the best way. We show here the effect of the two methods:

```
\score {
  \relative c'' {
    % Visible tempo marking
    \tempo 4=120
    a4 a a
    \once \hide Score.MetronomeMark
    % Invisible tempo marking to lengthen fermata in MIDI
    \tempo 4=80
    a4\fermata |
    % New tempo for next section
    \tempo 4=100
    a4 a a a |
  }
  \layout { }
  \midi { }
}
```



```
\score {
  \relative c'' {
    % Visible tempo marking
    \tempo 4=120
    a4 a a
    \once \omit Score.MetronomeMark
    % Invisible tempo marking to lengthen fermata in MIDI
    \tempo 4=80
    a4\fermata |
    % New tempo for next section
    \tempo 4=100
    a4 a a a |
  }
  \layout { }
  \midi { }
}
```



Both methods remove the metronome mark which lengthens the fermata from the printed output, and both affect the MIDI timing as required, but the transparent metronome mark in the first

line forces the following tempo indication too high while the second (with the stencil removed) does not.

Vegeu també

Music Glossary: *Secció “system” in Glossari musical.*

4.7.2 Using variables for layout adjustments

Override commands are often long and tedious to type, and they have to be absolutely correct. If the same overrides are to be used many times it may be worth defining variables to hold them.

Suppose we wish to emphasize certain words in lyrics by printing them in bold italics. The `\italic` and `\bold` commands only work within lyrics if they are embedded, together with the word or words to be modified, within a `\markup` block, which makes them tedious to enter. The need to embed the words themselves prevents their use in simple variables. As an alternative we can use `\override` and `\revert` commands?

```
\override Lyrics.LyricText.font-shape = #'italic
\override Lyrics.LyricText.font-series = #'bold
```

```
\revert Lyrics.LyricText.font-shape
\revert Lyrics.LyricText.font-series
```

These would also be extremely tedious to enter if there were many words requiring emphasis. But we *can* define these as two variables and use those to bracket the words to be emphasized. Another advantage of using variables for these overrides is that the spaces around the dot are not necessary, since they are not being interpreted in `\lyricmode` directly. Here’s an example of this, although in practice we would choose shorter names for the variables to make them quicker to type:

```
emphasize = {
  \override Lyrics.LyricText.font-shape = #'italic
  \override Lyrics.LyricText.font-series = #'bold
}

normal = {
  \revert Lyrics.LyricText.font-shape
  \revert Lyrics.LyricText.font-series
}

global = { \key c \major \time 4/4 \partial 4 }

SopranoMusic = \relative c' { c4 | e4. e8 g4 g | a4 a g }
AltoMusic = \relative c' { c4 | c4. c8 e4 e | f4 f e }
TenorMusic = \relative c { e4 | g4. g8 c4. b8 | a8 b c d e4 }
BassMusic = \relative c { c4 | c4. c8 c4 c | f8 g a b c4 }

VerseOne = \lyrics {
  E -- | ter -- nal \emphasize Fa -- ther, | \normal strong to save,
}

VerseTwo = \lyricmode {
  0 | \once \emphasize Christ, whose voice the | wa -- ters heard,
}

VerseThree = \lyricmode {
```

```

    0 | \emphasize Ho -- ly Spi -- rit, | \normal who didst brood
}

VerseFour = \lyricmode {
    0 | \emphasize Tri -- ni -- ty \normal of | love and pow'r
}

\score {
  \new ChoirStaff <<
    \new Staff <<
      \clef "treble"
      \new Voice = "Soprano" { \voiceOne \global \SopranoMusic }
      \new Voice = "Alto" { \voiceTwo \AltoMusic }
      \new Lyrics \lyricsto "Soprano" { \VerseOne }
      \new Lyrics \lyricsto "Soprano" { \VerseTwo }
      \new Lyrics \lyricsto "Soprano" { \VerseThree }
      \new Lyrics \lyricsto "Soprano" { \VerseFour }
    >>
    \new Staff <<
      \clef "bass"
      \new Voice = "Tenor" { \voiceOne \TenorMusic }
      \new Voice = "Bass" { \voiceTwo \BassMusic }
    >>
  >>
}

```

E - ter - nal ***Fa-ther***, strong to save,
 O ***Christ***, whose voice the wa - ters heard,
 O ***Ho - ly Spi-rit***, who didst brood
 O ***Tri - ni - ty*** of love and pow'r

4.7.3 Style sheets

The output that LilyPond produces can be heavily modified; see [Capítol 4 \[Tweaking output\]](#), [pàgina 88](#), for details. But what if you have many input files that you want to apply your tweaks to? Or what if you simply want to separate your tweaks from the actual music? This is quite easy to do.

Let's look at an example. Don't worry if you don't understand the parts with all the #(). This is explained in [Secció 4.7.5 \[Advanced tweaks with Scheme\]](#), [pàgina 147](#).

```

mpdolce =
  \tweak self-alignment-X #-0.6
  #(make-dynamic-script
    #{ \markup { \dynamic mp \normal-text \italic \bold dolce } #})

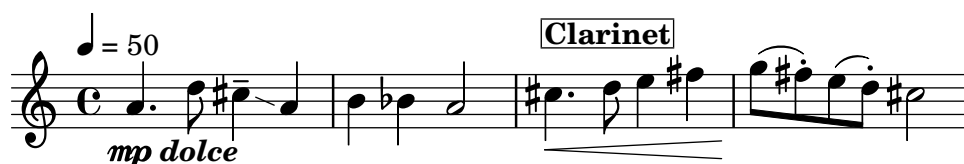
```

```

inst =
#(define-music-function
  (parser location string)
  (string?)
  #{ <>^\markup \bold \box #string #})

\relative c'' {
  \tempo 4=50
  a4.\mpdolce d8 cis4--\glissando a |
  b4 bes a2 |
  \inst "Clarinet"
  cis4.\< d8 e4 fis |
  g8(\! fis)-. e( d)-. cis2 |
}

```



Let's do something about the `mpdolce` and `inst` definitions. They produce the output we desire, but we might want to use them in another piece. We could simply copy-and-paste them at the top of every file, but that's an annoyance. It also leaves those definitions in our input files, and I personally find all the `#()` somewhat ugly. Let's hide them in another file:

```

%% save this to a file called "definitions.ily"
mpdolce =
  \tweak self-alignment-X #-0.6
  #(make-dynamic-script
    #{ \markup { \dynamic mp \normal-text \italic \bold dolce } #})

inst =
#(define-music-function
  (parser location string)
  (string?)
  #{ <>^\markup \bold \box #string #})

```

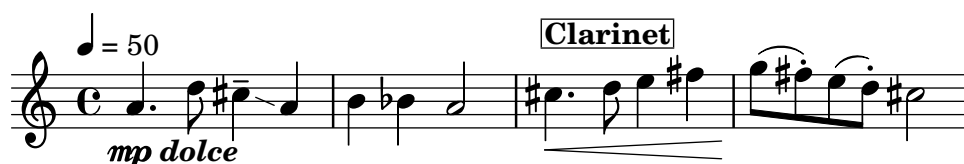
We will refer to this file using the `\include` command near the top of the music file. (The extension `.ily` is used to distinguish this included file, which is not meant to be compiled on its own, from the main file.) Now let's modify our music (let's save this file as `music.ly`).

```

\include "definitions.ily"

\relative c'' {
  \tempo 4=50
  a4.\mpdolce d8 cis4--\glissando a |
  b4 bes a2 |
  \inst "Clarinet"
  cis4.\< d8 e4 fis |
  g8(\! fis)-. e( d)-. cis2 |
}

```



That looks better, but let's make a few changes. The glissando is hard to see, so let's make it thicker and closer to the note heads. Let's put the metronome marking above the clef, instead of over the first note. And finally, my composition professor hates 'C' time signatures, so we'd better make that '4/4' instead.

Don't change 'music.ly', though. Replace our 'definitions.ily' with this:

```
%%% definitions.ily
mpdolce =
  \tweak self-alignment-X #-0.6
  #(make-dynamic-script
    #{ \markup { \dynamic mp \normal-text \italic \bold dolce } #})

inst =
  #(define-music-function
    (parser location string)
    (string?)
    #{ <>^\markup \bold \box #string #})

\layout{
  \context {
    \Score
    \override MetronomeMark.extra-offset = #'(-5 . 0)
    \override MetronomeMark.padding = #'3
  }
  \context {
    \Staff
    \override TimeSignature.style = #'numbered
  }
  \context {
    \Voice
    \override Glissando.thickness = #3
    \override Glissando.gap = #0.1
  }
}
```



That looks nicer! But now suppose that I want to publish this piece. My composition professor doesn't like 'C' time signatures, but I'm somewhat fond of them. Let's copy the current 'definitions.ily' to 'web-publish.ily' and modify that. Since this music is aimed at producing a pdf which will be displayed on the screen, we'll also increase the overall size of the output.

```
%%% web-publish.ily
mpdolce =
  \tweak self-alignment-X #-0.6
```

```

#(make-dynamic-script
  #{ \markup { \dynamic mp \normal-text \italic \bold dolce } #})

inst =
#(define-music-function
  (parser location string)
  (string?)
  #{ <>^\markup \bold \box #string #})

#(set-global-staff-size 23)

\layout{
  \context {
    \Score
    \override MetronomeMark.extra-offset = #'(-5 . 0)
    \override MetronomeMark.padding = #'3
  }
  \context {
    \Staff
  }
  \context {
    \Voice
    \override Glissando.thickness = #3
    \override Glissando.gap = #0.1
  }
}

```



Now in our music, I simply replace `\include "definitions.ily"` with `\include "web-publish.ily"`. Of course, we could make this even more convenient. We could make a ‘definitions.ily’ file which contains only the definitions of `mpdolce` and `inst`, a ‘web-publish.ily’ file which contains only the `\layout` section listed above, and a ‘university.ily’ file which contains only the tweaks to produce the output that my professor prefers. The top of ‘music.ly’ would then look like this:

```

\include "definitions.ily"

%%% Only uncomment one of these two lines!
\include "web-publish.ily"
%\include "university.ily"

```

This approach can be useful even if you are only producing one set of parts. I use half a dozen different ‘style sheet’ files for my projects. I begin every music file with `\include "../global.ily"`, which contains

```

%%%    global.ily
\version "2.19.16"

#(ly:set-option 'point-and-click #f)

\include "../init/init-defs.ly"
\include "../init/init-layout.ly"
\include "../init/init-headers.ly"
\include "../init/init-paper.ly"

```

4.7.4 Other sources of information

The Internals Reference documentation contains a lot of information about LilyPond, but even more information can be gathered by looking at the internal LilyPond files. To explore these, you must first find the directory appropriate to your system. The location of this directory depends (a) on whether you obtained LilyPond by downloading a precompiled binary from lilypond.org or whether you installed it from a package manager (i.e. distributed with GNU/Linux, or installed under fink or cygwin) or compiled it from source, and (b) on which operating system it is being used:

Downloaded from lilypond.org

- GNU/Linux

Navigate to

```
'INSTALLDIR/lilypond/usr/share/lilypond/current/'
```

- MacOS X

Navigate to

```
'INSTALLDIR/LilyPond.app/Contents/Resources/share/lilypond/current/'
```

by either `cd`-ing into this directory from the Terminal, or control-clicking on the LilyPond application and selecting 'Show Package Contents'.

- Windows

Using Windows Explorer, navigate to

```
'INSTALLDIR/LilyPond/usr/share/lilypond/current/'
```

Installed from a package manager or compiled from source

Navigate to '`PREFIX/share/lilypond/X.Y.Z/`', where *PREFIX* is set by your package manager or `configure` script, and *X.Y.Z* is the LilyPond version number.

Within this directory the two interesting subdirectories are

- '`ly/`' - contains files in LilyPond format
- '`scm/`' - contains files in Scheme format

Let's begin by looking at some files in '`ly/`'. Open '`ly/property-init.ly`' in a text editor. The one you normally use for `.ly` files will be fine. This file contains the definitions of all the standard LilyPond predefined commands, such as `\stemUp` and `\slurDotted`. You will see that these are nothing more than definitions of variables containing one or a group of `\override` commands. For example, `/tieDotted` is defined to be:

```

tieDotted = {
  \override Tie.dash-period = #0.75
  \override Tie.dash-fraction = #0.1
}

```

If you do not like the default values these predefined commands can be redefined easily, just like any other variable, at the head of your input file.

The following are the most useful files to be found in ‘ly/’:

Filename	Contents
‘ly/engraver-init.ly’	Definitions of engraver Contexts
‘ly/paper-defaults-init.ly’	Specifications of paper-related defaults
‘ly/performer-init.ly’	Definitions of performer Contexts
‘ly/property-init.ly’	Definitions of all common predefined commands
‘ly/spanner-init.ly’	Definitions of spanner-related predefined commands

Other settings (such as the definitions of markup commands) are stored as ‘.scm’ (Scheme) files. The Scheme programming language is used to provide a programmable interface into LilyPond internal operation. Further explanation of these files is currently outside the scope of this manual, as a knowledge of the Scheme language is required. Users should be warned that a substantial amount of technical knowledge or time is required to understand Scheme and these files (see [Secció “Scheme tutorial” in *Extendre*](#)).

If you have this knowledge, the Scheme files which may be of interest are:

Filename	Contents
‘scm/auto-beam.scm’	Sub-beaming defaults
‘scm/define-grobs.scm’	Default settings for grob properties
‘scm/define-markup-commands.scm’	Specify all markup commands
‘scm/midi.scm’	Default settings for MIDI output
‘scm/output-lib.scm’	Settings that affect appearance of frets, colors, accidentals, bar lines, etc
‘scm/parser-clef.scm’	Definitions of supported clefs
‘scm/script.scm’	Default settings for articulations

4.7.5 Advanced tweaks with Scheme

Although many things are possible with the `\override` and `\tweak` commands, an even more powerful way of modifying the action of LilyPond is available through a programmable interface to the LilyPond internal operation. Code written in the Scheme programming language can be incorporated directly in the internal operation of LilyPond. Of course, at least a basic knowledge of programming in Scheme is required to do this, and an introduction is provided in the [Secció “Scheme tutorial” in *Extendre*](#).

As an illustration of one of the many possibilities, instead of setting a property to a constant it can be set to a Scheme procedure which is then called whenever that property is accessed by LilyPond. The property can then be set dynamically to a value determined by the procedure at the time it is called. In this example we color the note head in accordance with its position on the staff.

```
#(define (color-notehead grob)
  "Color the notehead according to its position on the staff."
  (let ((mod-position (modulo (ly:grob-property grob 'staff-position)
                              7)))
    (case mod-position
      ;; Return rainbow colors
      ((1) (x11-color 'red )) ; for C
      ((2) (x11-color 'orange )) ; for D
      ((3) (x11-color 'yellow )) ; for E
      ((4) (x11-color 'green )) ; for F
```

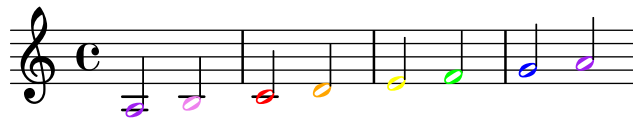


```

      ((5) (x11-color 'blue )) ; for G
      ((6) (x11-color 'purple )) ; for A
      ((0) (x11-color 'violet )) ; for B
    )))

\relative c' {
  % Arrange to obtain color from color-notehead procedure
  \override NoteHead.color = #color-notehead
  a2 b | c2 d | e2 f | g2 a |
}

```



Further examples showing the use of these programmable interfaces can be found in [Secció “Callback functions”](#) in *Extendre*.

Annex A Templates

This section of the manual contains templates with the LilyPond score already set up for you. Just add notes, run LilyPond, and enjoy beautiful printed scores!

A.1 Built-in templates

A template, suitable for a range of choral music, is built into LilyPond. This may be used to create simple choral music, with or without piano accompaniment, in two or four staves. Unlike other templates, this template is ‘built-in’, which means it does not need to be copied and edited: instead it is simply `\include`’d in the input file.

Nota: Unlike most included files, this built-in template must be `\include`’d at the *end* of the input file.

The required music expressions are entered by defining values for specific variables. These definition must come before the `\include`’d file.

The music may be set out with one or two voices per staff by setting `TwoVoicesPerStaff` to `##f` or `##t` respectively.

Here’s the complete input file for producing a full four-part SATB arrangement with individual lyrics and piano accompaniment:

```
SopranoMusic = \relative { a'4\f a8 a a4 a }
SopranoLyrics = \lyricmode { Sop -- ra -- no ly -- rics }
AltoMusic = \relative { d'4\f d d d }
AltoLyrics = \lyricmode { Al -- to ly -- rics }
TenorMusic = \relative { a4\p a a a }
TenorLyrics = \lyricmode { Te -- nor ly -- rics }
BassMusic = \relative { c2\p c4 c }
BassLyrics = \lyricmode { Bass ly -- rics }
PianoRHMus = \relative { c' e g c }
PianoDynamics = { s2\mp s4 s4 }
PianoLHMus = \relative { c e g c }
\include "satb.ly"
```



The same input can be used to produce a score with two voices per staff just by setting `TwoVoicesPerStaff` to `##t`. Again, each voice has individual lyrics.

```
SopranoMusic = \relative { a'4\f a8 a a4 a }
SopranoLyrics = \lyricmode { Sop -- ra -- no ly -- rics }
AltoMusic = \relative { d'4\f d d d }
AltoLyrics = \lyricmode { Al -- to ly -- rics }
TenorMusic = \relative { a4\p a a a }
TenorLyrics = \lyricmode { Te -- nor ly -- rics }
BassMusic = \relative { c2\p c4 c }
BassLyrics = \lyricmode { Bass ly -- rics }
```

```
PianoRHMusical = \relative { c' e g c }
PianoDynamics = { s2\mp s4 s4 }
PianoLHMusical = \relative { c e g c }
TwoVoicesPerStaff = ##t
\include "satb.ly"
```



When `TwoVoicesPerStaff` is set to false or allowed to default, any of the music variables may be omitted to produce arrangements with fewer voices. Here, for example, is how the input file for a Soprano/Bass duet might be written:

```
SopranoMusical = \relative { c' c c c }
SopranoLyrics = \lyricmode { High voice ly -- rics }
BassMusical = \relative { a a a a }
BassLyrics = \lyricmode { Low voice ly -- rics }
\include "satb.ly"
```

A second verse or alternative lyrics may be added to each of the parts:

```
SopranoMusical = \relative { a'4 a a a }
SopranoLyricsOne = \lyricsto "Soprano" {
  \set stanza = "1."
  Words to verse one
}
SopranoLyricsTwo = \lyricsto "Soprano" {
  \set stanza = "2."
  Words to verse two
}
\include "satb.ly"
```

When the lyrics and rhythms are the same for every part, the vocal music is best arranged on two staves with two voices in each. Up to nine verses may be provided. Here's an unaccompanied example with just three verses.

```
SopranoMusical = \relative { a' a a a }
AltoMusical = \relative { f' f f f }
VerseOne = \lyricmode {
  \set stanza = "1."
  Words to verse one
}
VerseTwo = \lyricmode {
  \set stanza = "2."
  Words to verse two
}
```

```

VerseThree = \lyricmode {
  \set stanza = "3."
  Words to verse three
}
TenorMusic = \relative { a a a a }
BassMusic = \relative { f f f f }
TwoVoicesPerStaff = ##t
\include "satb.ly"

```

Other variables may be given values. The key signature and the time signature may be changed from the default:

```

Key = \key a \major
Time = {
  \time 5/4
  \tempo "Allegro" 4 = 144
}
SopranoMusic = \relative { gis' gis gis gis gis }
AltoMusic = \relative { cis' cis cis cis cis }
VerseOne = \lyricmode { Words to this du -- et }
TwoVoicesPerStaff = ##t
\include "satb.ly"

```

Allegro (♩ = 144)

The instrument names and/or the short instrument names may be changed:

```

SopranoMusic = \relative { c' c c c }
SopranoLyrics = \lyricmode { High voice ly -- rics }
SopranoInstrumentName = "Soprano 1"
SopranoShortInstrumentName = "S1"
AltoMusic = \relative { a' a a a }
AltoLyrics = \lyricmode { Low voice ly -- rics }
AltoInstrumentName = "Soprano 2"
AltoShortInstrumentName = "S2"
\include "satb.ly"

```

A descant may be added by defining values for the variable `DescantMusic` and descant lyrics may be provided by defining values for `DescantLyrics`.

`\header` and `\paper` blocks may be added as normal. A `\layout` block may be provided as the value of the `Layout` variable:

```
Layout = \layout { ... }
```

The complete set of variables which may be changed can be seen by examining the file ‘ly/satb.ly’.

Vegeu també

Learning Manual: [\[Organizing pieces with variables\]](#), pàgina [\[undefined\]](#), Secció A.5 [\[Vocal ensembles templates\]](#), pàgina 161, [\[undefined\]](#) [\[Extending the templates\]](#), pàgina [\[undefined\]](#).

Advertiments i problemes coneguts

More complex arrangements of SATB choral music are not possible with these simple built-in templates.

A.2 Single staff templates

A.2.1 Notes only

This very simple template gives you a staff with notes, suitable for a solo instrument or a melodic fragment. Cut and paste this into a file, add notes, and you're finished!

```
\version "2.19.16"
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

\score {
  \new Staff \melody
  \layout { }
  \midi { }
}
```



A.2.2 Notes and lyrics

This small template demonstrates a simple melody with lyrics. Cut and paste, add notes, then words for the lyrics. This example turns off automatic beaming, which is common for vocal parts. To use automatic beaming, change or comment out the relevant line.

```
\version "2.19.16"
melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

\score{
  <<
  \new Voice = "one" {
    \autoBeamOff
```

```

        \melody
      }
      \new Lyrics \lyricsto "one" \text
    >>
    \layout { }
    \midi { }
  }

```



A.2.3 Notes and chords

Want to prepare a lead sheet with a melody and chords? Look no further!

```

melody = \relative c' {
  \clef treble
  \key c \major
  \time 4/4

  f4 e8[ c] d4 g
  a2 ~ a
}

harmonies = \chordmode {
  c4:m f:min7 g:maj c:aug
  d2:dim b:sus
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Staff \melody
  >>
  \layout{ }
  \midi { }
}

```



A.2.4 Notes, lyrics, and chords

This template allows the preparation of a song with melody, words, and chords.

```

melody = \relative c' {
  \clef treble
  \key c \major

```

```

\time 4/4

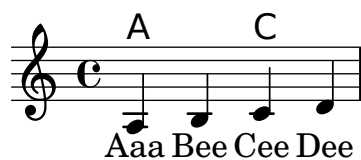
a4 b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

harmonies = \chordmode {
  a2 c
}

\score {
  <<
    \new ChordNames {
      \set chordChanges = ##t
      \harmonies
    }
    \new Voice = "one" { \autoBeamOff \melody }
    \new Lyrics \lyricsto "one" \text
  >>
  \layout { }
  \midi { }
}

```



A.3 Piano templates

A.3.1 Solo piano

Here is a simple piano staff with some notes.

```

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

```

```

\score {
  \new PianoStaff <<
    \set PianoStaff.instrumentName = #"Piano  "
    \new Staff = "upper" \upper
    \new Staff = "lower" \lower
  >>
  \layout { }
  \midi { }
}

```



A.3.2 Piano and melody with lyrics

Here is a typical song format: one staff with the melody and lyrics, with piano accompaniment underneath.

```

melody = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a b c d
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

\score {
  <<
    \new Voice = "mel" { \autoBeamOff \melody }

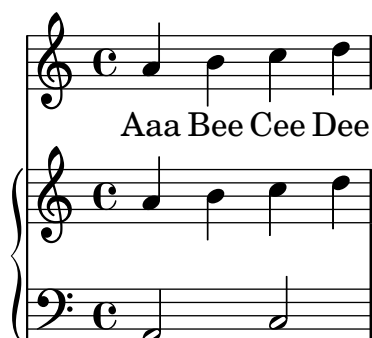
```



```

\new Lyrics \lyricsto mel \text
\new PianoStaff <<
  \new Staff = "upper" \upper
  \new Staff = "lower" \lower
  >>
>>
\layout {
  \context { \Staff \RemoveEmptyStaves }
}
\midi { }
}

```



A.3.3 Piano centered lyrics

Instead of having a full staff for the melody and lyrics, lyrics can be centered between the staves of a piano staff.

```

upper = \relative c'' {
  \clef treble
  \key c \major
  \time 4/4

  a4 b c d
}

lower = \relative c {
  \clef bass
  \key c \major
  \time 4/4

  a2 c
}

text = \lyricmode {
  Aaa Bee Cee Dee
}

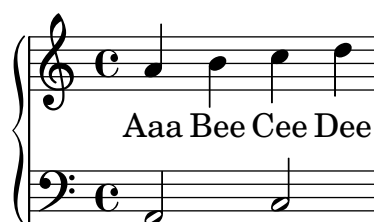
\score {
  \new GrandStaff <<
    \new Staff = upper { \new Voice = "singer" \upper }
    \new Lyrics \lyricsto "singer" \text
    \new Staff = lower { \lower }
  >>
}

```

```

\layout {
  \context {
    \GrandStaff
    \accepts "Lyrics"
  }
  \context {
    \Lyrics
    \consists "Bar_engraver"
  }
}
\midi { }
}

```



A.4 String quartet templates

A.4.1 String quartet

This template demonstrates a simple string quartet. It also uses a `\global` section for time and key signatures

```

global= {
  \time 4/4
  \key c \major
}

violinOne = \new Voice \relative c'' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d
  e1

  \bar "|"
}

violinTwo = \new Voice \relative c'' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f
  e1

  \bar "|"
}

viola = \new Voice \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto
}

```

```

e2 d
c1

\bar "|"
}

cello = \new Voice \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass

  c2 b
  a1

  \bar "|"
}

\score {
  \new StaffGroup <<
    \new Staff << \global \violinOne >>
    \new Staff << \global \violinTwo >>
    \new Staff << \global \viola >>
    \new Staff << \global \cello >>
  >>
  \layout { }
  \midi { }
}

```

Violin 1

Violin 2

Viola

Cello

A.4.2 String quartet parts

The “String quartet template” snippet produces a nice string quartet, but what if you needed to print parts? This new template demonstrates how to use the `\tag` feature to easily split a piece into individual parts.

You need to split this template into separate files; the filenames are contained in comments at the beginning of each file. `piece.ly` contains all the music definitions. The other files – `score.ly`, `vn1.ly`, `vn2.ly`, `vla.ly`, and `vlc.ly` – produce the appropriate part.

Do not forget to remove specified comments when using separate files!

```

%%%% piece.ly

```

```

%%%% (This is the global definitions file)

global= {
  \time 4/4
  \key c \major
}

Violinone = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 1 "

  c2 d e1

  \bar "|" } } %*****
Violintwo = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Violin 2 "

  g2 f e1

  \bar "|" } } %*****
Viola = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Viola "
  \clef alto

  e2 d c1

  \bar "|" } } %*****
Cello = \new Voice { \relative c' {
  \set Staff.instrumentName = #"Cello "
  \clef bass

  c2 b a1

  \bar "|." } } %*****

music = {
  <<
    \tag #'score \tag #'vn1 \new Staff { << \global \Violinone >> }
    \tag #'score \tag #'vn2 \new Staff { << \global \Violintwo>> }
    \tag #'score \tag #'vla \new Staff { << \global \Viola>> }
    \tag #'score \tag #'vlc \new Staff { << \global \Cello>> }
  >>
}

%%% These are the other files you need to save on your computer

%%%%% score.ly
%%%%% (This is the main file)

%%% uncomment the line below when using a separate file
%\include "piece.ly"
#(set-global-staff-size 14)
\score {

```

```

    \new StaffGroup \keepWithTag #'score \music
    \layout { }
    \midi { }
}

%{ Uncomment this block when using separate files

%%%% vn1.ly
%%%% (This is the Violin 1 part file)

\include "piece.ly"
\score {
    \keepWithTag #'vn1 \music
    \layout { }
}

%%%% vn2.ly
%%%% (This is the Violin 2 part file)

\include "piece.ly"
\score {
    \keepWithTag #'vn2 \music
    \layout { }
}

%%%% vla.ly
%%%% (This is the Viola part file)

\include "piece.ly"
\score {
    \keepWithTag #'vla \music
    \layout { }
}

%%%% vlc.ly
%%%% (This is the Cello part file)

\include "piece.ly"
\score {
    \keepWithTag #'vlc \music
    \layout { }
}

%}

```



A.5 Vocal ensembles templates

The templates shown below should be copied into your score and edited there. If you have a relatively simple SATB layout you may prefer to use the built-in templates, which can simply be `\include'd`, see [Secció A.1 \[Built-in templates\]](#), [pàgina 149](#).

A.5.1 SATB vocal score

Here is a standard four-part SATB vocal score. With larger ensembles, it is often useful to include a section which is included in all parts. For example, the time signature and key signature are almost always the same for all parts. Like in the “Hymn” template, the four voices are regrouped on only two staves.

```
\paper {
  top-system-spacing.basic-distance = #10
  score-system-spacing.basic-distance = #20
  system-system-spacing.basic-distance = #20
  last-bottom-spacing.basic-distance = #10
}
```

```
global = {
  \key c \major
  \time 4/4
}
```

```
sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
```

```
sopWords = \lyricmode {
  hi hi hi hi
}
```

```
altoMusic = \relative c' {
  e4 f d e
}
```

```
altoWords = \lyricmode {
  ha ha ha ha
}
```

```
tenorMusic = \relative c' {
  g4 a f g
}
```

```
tenorWords = \lyricmode {
  hu hu hu hu
}
```

```

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

\score {
  \new ChoirStaff <<
    \new Lyrics = "sopranos" \with {
      % this is needed for lyrics above a staff
      \override VerticalAxisGroup.staff-affinity = #DOWN
    }
    \new Staff = "women" <<
      \new Voice = "sopranos" {
        \voiceOne
        << \global \sopMusic >>
      }
      \new Voice = "altos" {
        \voiceTwo
        << \global \altoMusic >>
      }
    >>
    \new Lyrics = "altos"
    \new Lyrics = "tenors" \with {
      % this is needed for lyrics above a staff
      \override VerticalAxisGroup.staff-affinity = #DOWN
    }
    \new Staff = "men" <<
      \clef bass
      \new Voice = "tenors" {
        \voiceOne
        << \global \tenorMusic >>
      }
      \new Voice = "basses" {
        \voiceTwo << \global \bassMusic >>
      }
    >>
    \new Lyrics = "basses"
    \context Lyrics = "sopranos" \lyricsto "sopranos" \sopWords
    \context Lyrics = "altos" \lyricsto "altos" \altoWords
    \context Lyrics = "tenors" \lyricsto "tenors" \tenorWords
    \context Lyrics = "basses" \lyricsto "basses" \bassWords
  >>
}

```



A.5.2 SATB vocal score and automatic piano reduction

This template adds an automatic piano reduction to the standard SATB vocal score demonstrated in “Vocal ensemble template”. This demonstrates one of the strengths of LilyPond – you can use a music definition more than once. If any changes are made to the vocal notes (say, `tenorMusic`), then the changes will also apply to the piano reduction.

```
\paper {
  top-system-spacing.basic-distance = #10
  score-system-spacing.basic-distance = #20
  system-system-spacing.basic-distance = #20
  last-bottom-spacing.basic-distance = #10
}
```

```
global = {
  \key c \major
  \time 4/4
}
```

```
sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}
```

```
altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}
```

```
tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}
```

```
bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
```



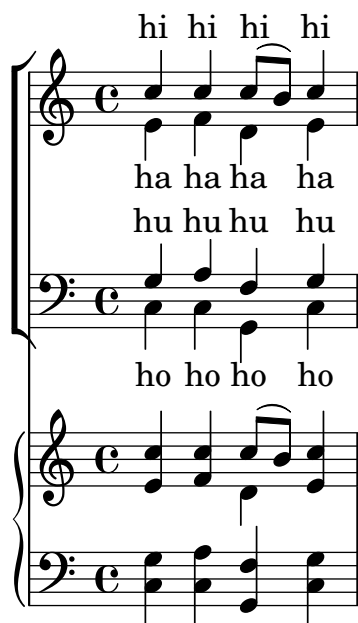
```

    ho ho ho ho
}

\score {
  <<
    \new ChoirStaff <<
      \new Lyrics = "sopranos" \with {
        % This is needed for lyrics above a staff
        \override VerticalAxisGroup.staff-affinity = #DOWN
      }
      \new Staff = "women" <<
        \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
        \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
      >>
      \new Lyrics = "altos"
      \new Lyrics = "tenors" \with {
        % This is needed for lyrics above a staff
        \override VerticalAxisGroup.staff-affinity = #DOWN
      }

      \new Staff = "men" <<
        \clef bass
        \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
        \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
      >>
      \new Lyrics = "basses"
      \context Lyrics = "sopranos" \lyricsto "sopranos" \sopWords
      \context Lyrics = "altos" \lyricsto "altos" \altoWords
      \context Lyrics = "tenors" \lyricsto "tenors" \tenorWords
      \context Lyrics = "basses" \lyricsto "basses" \bassWords
    >>
    \new PianoStaff <<
      \new Staff <<
        \set Staff.printPartCombineTexts = ##f
        \partcombine
        << \global \sopMusic >>
        << \global \altoMusic >>
      >>
      \new Staff <<
        \clef bass
        \set Staff.printPartCombineTexts = ##f
        \partcombine
        << \global \tenorMusic >>
        << \global \bassMusic >>
      >>
    >>
  >>
}

```



A.5.3 SATB with aligned contexts

This template is basically the same as the simple “Vocal ensemble” template, with the exception that here all the lyrics lines are placed using `alignAboveContext` and `alignBelowContext`.

```

global = {
  \key c \major
  \time 4/4
}

sopMusic = \relative c'' {
  c4 c c8[( b)] c4
}
sopWords = \lyricmode {
  hi hi hi hi
}

altoMusic = \relative c' {
  e4 f d e
}
altoWords = \lyricmode {
  ha ha ha ha
}

tenorMusic = \relative c' {
  g4 a f g
}
tenorWords = \lyricmode {
  hu hu hu hu
}

bassMusic = \relative c {
  c4 c g c
}
bassWords = \lyricmode {
  ho ho ho ho
}

```

```

\score {
  \new ChoirStaff <<
    \new Staff = "women" <<
      \new Voice = "sopranos" { \voiceOne << \global \sopMusic >> }
      \new Voice = "altos" { \voiceTwo << \global \altoMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = #"women" }
      \lyricsto "sopranos" \sopWords
    \new Lyrics \with { alignBelowContext = #"women" }
      \lyricsto "altos" \altoWords
    % we could remove the line about this with the line below, since
    % we want the alto lyrics to be below the alto Voice anyway.
    % \new Lyrics \lyricsto "altos" \altoWords

    \new Staff = "men" <<
      \clef bass
      \new Voice = "tenors" { \voiceOne << \global \tenorMusic >> }
      \new Voice = "basses" { \voiceTwo << \global \bassMusic >> }
    >>
    \new Lyrics \with { alignAboveContext = #"men" }
      \lyricsto "tenors" \tenorWords
    \new Lyrics \with { alignBelowContext = #"men" }
      \lyricsto "basses" \bassWords
    % again, we could replace the line above this with the line below.
    % \new Lyrics \lyricsto "basses" \bassWords
  >>
}

```



A.5.4 SATB on four staves

SATB choir template (four staves)

```

global = {
  \key c \major
  \time 4/4
  \dynamicUp
}
sopranonotes = \relative c'' {
  c2 \p \< d c d \f
}
sopranowords = \lyricmode { do do do do }

```

```

altonotes = \relative c'' {
  c2\p d c d
}
altowords = \lyricmode { re re re re }
tenornotes = {
  \clef "G_8"
  c2\mp d c d
}
tenorwords = \lyricmode { mi mi mi mi }
bassnotes = {
  \clef bass
  c2\mf d c d
}
basswords = \lyricmode { mi mi mi mi }

\score {
  \new ChoirStaff <<
    \new Staff <<
      \new Voice = "soprano" <<
        \global
        \sopranonotes
      >>
      \new Lyrics \lyricsto "soprano" \sopranowords
    >>
    \new Staff <<
      \new Voice = "alto" <<
        \global
        \altonotes
      >>
      \new Lyrics \lyricsto "alto" \altowords
    >>
    \new Staff <<
      \new Voice = "tenor" <<
        \global
        \tenornotes
      >>
      \new Lyrics \lyricsto "tenor" \tenorwords
    >>
    \new Staff <<
      \new Voice = "bass" <<
        \global
        \bassnotes
      >>
      \new Lyrics \lyricsto "bass" \basswords
    >>
  >>
}

```



A.5.5 Solo verse and two-part refrain

This template creates a score which starts with a solo verse and continues into a refrain for two voices. It also demonstrates the use of spacer rests within the `\global` variable to define meter changes (and other elements common to all parts) throughout the entire score.

```

global = {
  \key g \major

  % verse
  \time 3/4
  s2.*2
  \break

  % refrain
  \time 2/4
  s2*2
  \bar "|"
}

SoloNotes = \relative g' {
  \clef "treble"

  % verse
  g4 g g |
  b4 b b |

  % refrain
  R2*2 |
}

SoloLyrics = \lyricmode {
  One two three |
  four five six |
}

SopranoNotes = \relative c' {

```

```

\clef "treble"

% verse
R2.*2 |

% refrain
c4 c |
g4 g |
}

SopranoLyrics = \lyricmode {
  la la |
  la la |
}

BassNotes = \relative c {
  \clef "bass"

  % verse
  R2.*2 |

  % refrain
  c4 e |
  d4 d |
}

BassLyrics = \lyricmode {
  dum dum |
  dum dum |
}

\score {
  <<
    \new Voice = "SoloVoice" << \global \SoloNotes >>
    \new Lyrics \lyricsto "SoloVoice" \SoloLyrics

    \new ChoirStaff <<
      \new Voice = "SopranoVoice" << \global \SopranoNotes >>
      \new Lyrics \lyricsto "SopranoVoice" \SopranoLyrics

      \new Voice = "BassVoice" << \global \BassNotes >>
      \new Lyrics \lyricsto "BassVoice" \BassLyrics
    >>
  >>
  \layout {
    ragged-right = ##t
    \context { \Staff
      % these lines prevent empty staves from being printed
      \RemoveEmptyStaves
      \override VerticalAxisGroup.remove-first = ##t
    }
  }
}

```

}



A.5.6 Hymn tunes

This code shows one way of setting out a hymn tune when each line starts and ends with a partial measure. It also shows how to add the verses as stand-alone text under the music.

```
Timeline = {
  \time 4/4
  \tempo 4=96
  \partial 2
  s2 | s1 | s2 \breathe s2 | s1 | s2 \bar "||" \break
  s2 | s1 | s2 \breathe s2 | s1 | s2 \bar "||"
}

SopranoMusic = \relative g' {
  g4 g | g g g g | g g g g | g g g g | g2
  g4 g | g g g g | g g g g | g g g g | g2
}

AltoMusic = \relative c' {
  d4 d | d d d d | d d d d | d d d d | d2
  d4 d | d d d d | d d d d | d d d d | d2
}

TenorMusic = \relative a {
  b4 b | b b b b | b b b b | b b b b | b2
  b4 b | b b b b | b b b b | b b b b | b2
}

BassMusic = \relative g {
  g4 g | g g g g | g g g g | g g g g | g2
  g4 g | g g g g | g g g g | g g g g | g2
}

global = {
  \key g \major
}

\score { % Start score
```

```

<<
  \new PianoStaff << % Start pianostaff
    \new Staff << % Start Staff = RH
      \global
      \clef "treble"
      \new Voice = "Soprano" << % Start Voice = "Soprano"
        \Timeline
        \voiceOne
        \SopranoMusic
      >> % End Voice = "Soprano"
      \new Voice = "Alto" << % Start Voice = "Alto"
        \Timeline
        \voiceTwo
        \AltoMusic
      >> % End Voice = "Alto"
    >> % End Staff = RH
  \new Staff << % Start Staff = LH
    \global
    \clef "bass"
    \new Voice = "Tenor" << % Start Voice = "Tenor"
      \Timeline
      \voiceOne
      \TenorMusic
    >> % End Voice = "Tenor"
    \new Voice = "Bass" << % Start Voice = "Bass"
      \Timeline
      \voiceTwo
      \BassMusic
    >> % End Voice = "Bass"
  >> % End Staff = LH
>> % End pianostaff
>>
} % End score

\markup {
  \fill-line {
    ""
    {
      \column {
        \left-align {
          "This is line one of the first verse"
          "This is line two of the same"
          "And here's line three of the first verse"
          "And the last line of the same"
        }
      }
    }
  }
  ""
}

\paper { % Start paper block

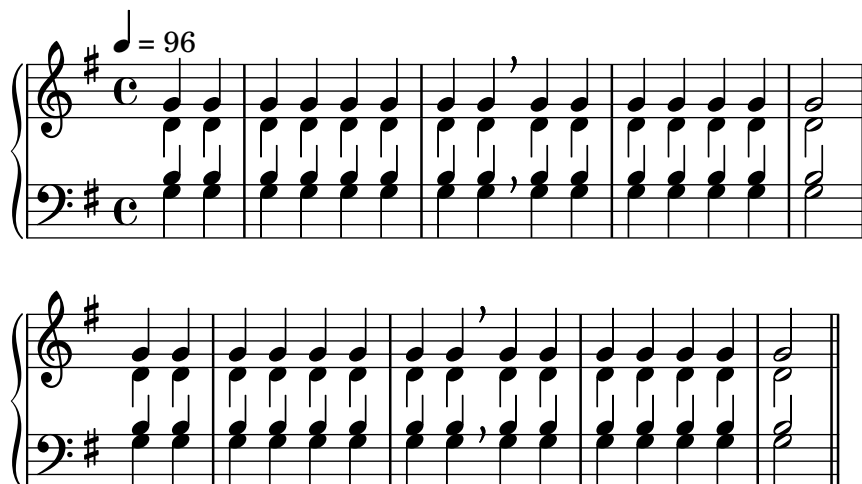
```



```

indent = 0      % don't indent first system
line-width = 130 % shorten line length to suit music
} % End paper block

```



This is line one of the first verse
 This is line two of the same
 And here's line three of the first verse
 And the last line of the same

A.5.7 Psalms

This template shows one way of setting out an Anglican psalm chant. It also shows how the verses may be added as stand-alone text under the music. The two verses are coded in different styles to demonstrate more possibilities.

```

SopranoMusic = \relative g' {
  g1 | c2 b | a1 | \bar "||"
  a1 | d2 c | c b | c1 | \bar "||"
}

```

```

AltoMusic = \relative c' {
  e1 | g2 g | f1 |
  f1 | f2 e | d d | e1 |
}

```

```

TenorMusic = \relative a {
  c1 | c2 c | c1 |
  d1 | g,2 g | g g | g1 |
}

```

```

BassMusic = \relative c {
  c1 | e2 e | f1 |
  d1 | b2 c | g' g | c,1 |
}

```

```

global = {
  \time 2/2
}

```

```

dot = \markup {
  \raise #0.7 \musicglyph #"dots.dot"
}

tick = \markup {
  \raise #1 \fontsize #-5 \musicglyph #"scripts.rvarcomma"
}

% Use markup to center the chant on the page
\markup {
  \fill-line {
    \score { % centered
      <<
        \new ChoirStaff <<
          \new Staff <<
            \global
            \clef "treble"
            \new Voice = "Soprano" <<
              \voiceOne
              \SopranoMusic
            >>
            \new Voice = "Alto" <<
              \voiceTwo
              \AltoMusic
            >>
          >>
        \new Staff <<
          \clef "bass"
          \global
          \new Voice = "Tenor" <<
            \voiceOne
            \TenorMusic
          >>
          \new Voice = "Bass" <<
            \voiceTwo
            \BassMusic
          >>
        >>
      >>
    }
  }
  \layout {
    \context {
      \Score
      \override SpacingSpanner.base-shortest-duration = #(ly:make-moment 1/2)
    }
    \context {
      \Staff
      \remove "Time_signature_engraver"
    }
  }
} % End score

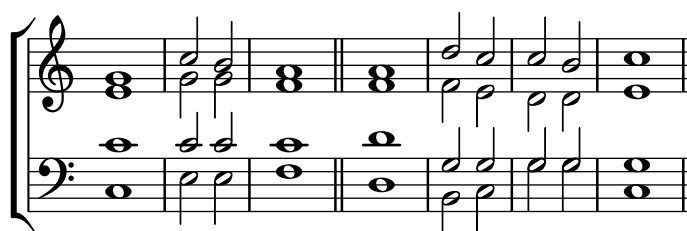
```

```

    }
} % End markup

\markup {
  \fill-line {
    \column {
      \left-align {
        \null \null \null
        \line {
          \fontsize #5 0
          \fontsize #3 come
          let us \bold sing | unto \dot the | Lord : let
        }
        \line {
          us heartily
          \concat { re \bold joice }
          in the | strength of | our
        }
        \line {
          sal | vation.
        }
        \null
        \line {
          \hspace #2.5 8. Today if ye will hear his voice *
        }
        \line {
          \concat { \bold hard en }
          \tick not your \tick hearts : as in the pro-
        }
        \line {
          vocation * and as in the \bold day of tempt- \tick
        }
        \line {
          -ation \tick in the \tick wilderness.
        }
      }
    }
  }
}

```



O come let us **sing** | unto • the | Lord : let
us heartily **rejoice** in the | strength of | our
sal | vation.

8. Today if ye will hear his voice *
harden ' not your ' hearts : as in the pro-
vocation * and as in the **day** of tempt- '
-ation ' in the ' wilderness.

A.6 Orchestral templates

A.6.1 Orchestra, choir and piano

This template demonstrates the use of nested `StaffGroup` and `GrandStaff` contexts to subgroup instruments of the same type together, and a way to use `\transpose` so that variables hold music for transposing instruments at concert pitch.

```

#(set-global-staff-size 17)
\paper {
  indent = 3.0\cm % space for instrumentName
  short-indent = 1.5\cm % space for shortInstrumentName
}

fluteMusic = \relative c' { \key g \major g'1 b }
% Pitches as written on a manuscript for Clarinet in A
% are transposed to concert pitch.
clarinetMusic = \transpose c' a
  \relative c'' { \key bes \major bes1 d }
trumpetMusic = \relative c { \key g \major g''1 b }
% Key signature is often omitted for horns
hornMusic = \transpose c' f
  \relative c { d'1 fis }
percussionMusic = \relative c { \key g \major g1 b }
sopranoMusic = \relative c'' { \key g \major g'1 b }
sopranoLyrics = \lyricmode { Lyr -- ics }
altoIMusic = \relative c' { \key g \major g'1 b }
altoIIMusic = \relative c' { \key g \major g'1 b }
altoILyrics = \sopranoLyrics
altoIIILyrics = \lyricmode { Ah -- ah }
tenorMusic = \relative c' { \clef "treble_8" \key g \major g1 b }
tenorLyrics = \sopranoLyrics
pianoRHMus = \relative c { \key g \major g''1 b }
pianoLHMus = \relative c { \clef bass \key g \major g1 b }
violinIMusic = \relative c' { \key g \major g'1 b }
violinIIMusic = \relative c' { \key g \major g'1 b }
violaMusic = \relative c { \clef alto \key g \major g'1 b }
celloMusic = \relative c { \clef bass \key g \major g1 b }
bassMusic = \relative c { \clef "bass_8" \key g \major g,1 b }

\score {
  <<
    \new StaffGroup = "StaffGroup_woodwinds" <<

```

```

\new Staff = "Staff_flute" {
  \set Staff.instrumentName = #"Flute"
  % shortInstrumentName, midiInstrument, etc.
  % may be set here as well
  \fluteMusic
}
\new Staff = "Staff_clarinet" {
  \set Staff.instrumentName =
  \markup { \concat { "Clarinet in B" \flat } }
  % Declare that written Middle C in the music
  % to follow sounds a concert B flat, for
  % output using sounded pitches such as MIDI.
  \transposition bes
  % Print music for a B-flat clarinet
  \transpose bes c' \clarinetMusic
}
>>
\new StaffGroup = "StaffGroup_brass" <<
  \new Staff = "Staff_hornI" {
    \set Staff.instrumentName = #"Horn in F"
    \transposition f
    \transpose f c' \hornMusic
  }
  \new Staff = "Staff_trumpet" {
    \set Staff.instrumentName = #"Trumpet in C"
    \trumpetMusic
  }
>>
\new RhythmicStaff = "RhythmicStaff_percussion" <<
  \set RhythmicStaff.instrumentName = #"Percussion"
  \percussionMusic
>>
\new PianoStaff <<
  \set PianoStaff.instrumentName = #"Piano"
  \new Staff { \pianoRHMus }
  \new Staff { \pianoLHMus }
>>
\new ChoirStaff = "ChoirStaff_choir" <<
  \new Staff = "Staff_soprano" {
    \set Staff.instrumentName = #"Soprano"
    \new Voice = "soprano"
    \sopranoMusic
  }
  \new Lyrics \lyricsto "soprano" { \sopranoLyrics }
  \new GrandStaff = "GrandStaff_alto"
  \with { \accepts Lyrics } <<
    \new Staff = "Staff_altoI" {
      \set Staff.instrumentName = #"Alto I"
      \new Voice = "altoI"
      \altoIMusic
    }
    \new Lyrics \lyricsto "altoI" { \altoILyrics }
  >>

```

```

        \new Staff = "Staff_altoII" {
            \set Staff.instrumentName = #"Alto II"
            \new Voice = "altoII"
            \altoIIMusic
        }
        \new Lyrics \lyricsto "altoII" { \altoIILyrics }
    >>
    \new Staff = "Staff_tenor" {
        \set Staff.instrumentName = #"Tenor"
        \new Voice = "tenor"
        \tenorMusic
    }
    \new Lyrics \lyricsto "tenor" { \tenorLyrics }
>>
\new StaffGroup = "StaffGroup_strings" <<
    \new GrandStaff = "GrandStaff_violins" <<
        \new Staff = "Staff_violinI" {
            \set Staff.instrumentName = #"Violin I"
            \violinIMusic
        }
        \new Staff = "Staff_violinII" {
            \set Staff.instrumentName = #"Violin II"
            \violinIIMusic
        }
    >>
    \new Staff = "Staff_viola" {
        \set Staff.instrumentName = #"Viola"
        \violaMusic
    }
    \new Staff = "Staff_cello" {
        \set Staff.instrumentName = #"Cello"
        \celloMusic
    }
    \new Staff = "Staff_bass" {
        \set Staff.instrumentName = #"Double Bass"
        \bassMusic
    }
>>
>>
\layout { }
}

```

Flute

Clarinet in B \flat

Horn in F

Trumpet in C

Percussion

Piano

Soprano

Alto I

Alto II

Tenor

Violin I

Violin II

Viola

Cello

Double Bass

Lyr - ics

Lyr - ics

Ah - ah

Lyr - ics

8

A.7 Ancient notation templates

A.7.1 Transcription of mensural music

When transcribing mensural music, an incipit at the beginning of the piece is useful to indicate the original key and tempo. While today musicians are used to bar lines in order to faster recognize rhythmic patterns, bar lines were not yet invented during the period of mensural music; in fact, the meter often changed after every few notes. As a compromise, bar lines are often printed between the staves rather than on the staves.

```
global = {
  \set Score.skipBars = ##t

  % incipit
  \once \hide Score.SystemStartBracket
  % Set tight spacing
  \override Score.SpacingSpanner.spacing-increment = #1.0
  \key f \major
  \time 2/2
  \once \override Staff.TimeSignature.style = #'neomensural
  \override Voice.NoteHead.style = #'neomensural
```

```

\override Voice.Rest.style = #'neomensural
\set Staff.printKeyCancellation = ##f
\cadenzaOn % turn off bar lines
\skip 1*10
\once \override Staff.BarLine.transparent = ##f
\bar "||"
\skip 1*1 % need this extra \skip such that clef change comes
          % after bar line
\bar ""

% main
\cadenzaOff % turn bar lines on again
\once \override Staff.Clef.full-size-change = ##t
\set Staff.forceClef = ##t
\key g \major
\time 4/4
\override Voice.NoteHead.style = #'default
\override Voice.Rest.style = #'default

% Setting printKeyCancellation back to #t must not
% occur in the first bar after the incipit. Dto. for forceClef.
% Therefore, we need an extra \skip.
\skip 1*1
\set Staff.printKeyCancellation = ##t
\set Staff.forceClef = ##f

\skip 1*7 % the actual music

% let finis bar go through all staves
\override Staff.BarLine.transparent = ##f

% finis bar
\bar "|."
}

discantusNotes = {
  \transpose c' c'' {
    \set Staff.instrumentName = #"Discantus  "

    % incipit
    \clef "neomensural-c1"
    c'1. s2 % two bars
    \skip 1*8 % eight bars
    \skip 1*1 % one bar

    % main
    \clef "treble"
    d'2. d'4 |
    b e' d'2 |
    c'4 e'4.( d'8 c' b |
    a4) b a2 |
    b4.( c'8 d'4) c'4 |
  }
}

```



```

        \once \hide NoteHead c'1 |
        b\breve |
    }
}

discantusLyrics = \lyricmode {
    % incipit
    IV-

    % main
    Ju -- bi -- |
    la -- te De -- |
    o, om --
    nis ter -- |
    ra, -- om- |
    "... " |
    -us. |
}

altusNotes = {
    \transpose c' c'' {
        \set Staff.instrumentName = #"Altus "

        % incipit
        \clef "neomensural-c3"
        r1          % one bar
        f1. s2      % two bars
        \skip 1*7 % seven bars
        \skip 1*1 % one bar

        % main
        \clef "treble"
        r2 g2. e4 fis g | % two bars
        a2 g4 e |
        fis g4.( fis16 e fis4) |
        g1 |
        \once \hide NoteHead g1 |
        g\breve |
    }
}

altusLyrics = \lyricmode {
    % incipit
    IV-

    % main
    Ju -- bi -- la -- te | % two bars
    De -- o, om -- |
    nis ter -- ra, |
    "... " |
    -us. |
}

```

```

tenorNotes = {
  \transpose c' c' {
    \set Staff.instrumentName = #"Tenor  "

    % incipit
    \clef "neomensural-c4"
    r\longa  % four bars
    r\breve  % two bars
    r1       % one bar
    c'1. s2  % two bars
    \skip 1*1 % one bar
    \skip 1*1 % one bar

    % main
    \clef "treble_8"
    R1 |
    R1 |
    R1 |
    r2 d'2. d'4 b e' | % two bars
    \once \hide NoteHead e'1 |
    d'\breve |
  }
}

tenorLyrics = \lyricmode {
  % incipit
  IV-

  % main
  Ju -- bi -- la -- te | % two bars
  "... " |
  -us. |
}

bassusNotes = {
  \transpose c' c' {
    \set Staff.instrumentName = #"Bassus  "

    % incipit
    \clef "bass"
    r\maxima % eight bars
    f1. s2   % two bars
    \skip 1*1 % one bar

    % main
    \clef "bass"
    R1 |
    R1 |
    R1 |
    R1 |
    g2. e4 |
  }
}

```

```

        \once \hide NoteHead e1 |
        g\breve |
    }
}

bassusLyrics = \lyricmode {
    % incipit
    IV-

    % main
    Ju -- bi- |
    "... " |
    -us. |
}

\score {
    \new StaffGroup = choirStaff <<
        \new Voice =
            "discantusNotes" << \global \discantusNotes >>
        \new Lyrics =
            "discantusLyrics" \lyricsto discantusNotes { \discantusLyrics }
        \new Voice =
            "altusNotes" << \global \altusNotes >>
        \new Lyrics =
            "altusLyrics" \lyricsto altusNotes { \altusLyrics }
        \new Voice =
            "tenorNotes" << \global \tenorNotes >>
        \new Lyrics =
            "tenorLyrics" \lyricsto tenorNotes { \tenorLyrics }
        \new Voice =
            "bassusNotes" << \global \bassusNotes >>
        \new Lyrics =
            "bassusLyrics" \lyricsto bassusNotes { \bassusLyrics }
    >>
    \layout {
        \context {
            \Score

            % no bars in staves
            \hide BarLine

            % incipit should not start with a start delimiter
            \remove "System_start_delimiter_engraver"
        }
        \context {
            \Voice

            % no slurs
            \hide Slur

            % The command below can be commented out in
            % short scores, but especially for large scores you

```

% will typically yield better line breaking and improve
 % overall spacing if you do not comment the command out.

```
\remove "Forbid_line_break_engraver"
}
}
}
```

Discantus

Altus

Tenor

Bassus

IV-

IV-

IV-

IV-

IV-

2

la-te De - o, om - nister -

- bi-la-te De - o, om - nis ter -

Ju -

5

ra, om- ... -us.

ra, ... -us.

- bi-la-te ... -us.

Ju - bi- ... -us.

A.7.2 Gregorian transcription template

This example demonstrates how to do modern transcription of Gregorian music. Gregorian music has no measure, no stems; it uses only half and quarter note heads, and special marks, indicating rests of different length.

```
\include "gregorian.ly"

chant = \relative c' {
  \set Score.timing = ##f
  f4 a2 \divisioMinima
  g4 b a2 f2 \divisioMaior
  g4( f) f( g) a2 \finalis
}

verba = \lyricmode {
  Lo -- rem ip -- sum do -- lor sit a -- met
}

\score {
  \new Staff <<
    \new Voice = "melody" \chant
    \new Lyrics = "one" \lyricsto melody \verba
  >>
  \layout {
    \context {
      \Staff
      \remove "Time_signature_engraver"
      \remove "Bar_engraver"
      \hide Stem
    }
    \context {
      \Voice
      \override Stem.length = #0
    }
    \context {
      \Score
      barAlways = ##t
    }
  }
}
```



A.8 Other templates

A.8.1 Jazz combo

This is quite an advanced template, for a jazz ensemble. Note that all instruments are notated in `\key c \major`. This refers to the key in concert pitch; the key will be automatically transposed if the music is within a `\transpose` section.

```

\header {
  title = "Song"
  subtitle = "(tune)"
  composer = "Me"
  meter = "moderato"
  piece = "Swing"
  tagline = \markup {
    \column {
      "LilyPond example file by Amelie Zapf,"
      "Berlin 07/07/2003"
    }
  }
}

%#(set-global-staff-size 16)
\include "english.ly"

%%%%%%%%%%%%%% Some macros %%%%%%%%%%%%%%%

sl = {
  \override NoteHead.style = #'slash
  \hide Stem
}
nsl = {
  \revert NoteHead.style
  \undo \hide Stem
}
crOn = \override NoteHead.style = #'cross
crOff = \revert NoteHead.style

%% insert chord name style stuff here.

jazzChords = { }

%%%%%%%%%%%%%% Keys'n'things %%%%%%%%%%%%%%%

global = { \time 4/4 }

Key = { \key c \major }

% ##### Horns #####

% ----- Trumpet -----
trpt = \transpose c d \relative c' {
  \Key
  c1 | c | c |
}
trpHarmony = \transpose c' d {
  \jazzChords
}
trumpet = {
  \global

```

```

\set Staff.instrumentName = #"Trumpet"
\clef treble
<<
  \trpt
>>
}

% ----- Alto Saxophone -----
alto = \transpose c a \relative c' {
  \Key
  c1 | c | c |
}
altoHarmony = \transpose c' a {
  \jazzChords
}
altoSax = {
  \global
  \set Staff.instrumentName = #"Alto Sax"
  \clef treble
  <<
    \alto
  >>
}

% ----- Baritone Saxophone -----
bari = \transpose c a' \relative c {
  \Key
  c1
  c1
  \sl
  d4^"Solo" d d d
  \nsl
}
bariHarmony = \transpose c' a \chordmode {
  \jazzChords s1 s d2:maj e:m7
}
bariSax = {
  \global
  \set Staff.instrumentName = #"Bari Sax"
  \clef treble
  <<
    \bari
  >>
}

% ----- Trombone -----
tbone = \relative c {
  \Key
  c1 | c | c |
}
tboneHarmony = \chordmode {
  \jazzChords
}

```

```

}
trombone = {
  \global
  \set Staff.instrumentName = #"Trombone"
  \clef bass
  <<
    \tbone
  >>
}

% ##### Rhythm Section #####

% ----- Guitar -----
gtr = \relative c' {
  \Key
  c1
  \sl
  b4 b b b
  \nsl
  c1
}
gtrHarmony = \chordmode {
  \jazzChords
  s1 c2:min7+ d2:maj9
}
guitar = {
  \global
  \set Staff.instrumentName = #"Guitar"
  \clef treble
  <<
    \gtr
  >>
}

%% ----- Piano -----
rhUpper = \relative c' {
  \voiceOne
  \Key
  c1 | c | c
}
rhLower = \relative c' {
  \voiceTwo
  \Key
  e1 | e | e
}

lhUpper = \relative c' {
  \voiceOne
  \Key
  g1 | g | g
}
lhLower = \relative c {

```



```

    \voiceTwo
    \Key
    c1 | c | c
}

PianoRH = {
    \clef treble
    \global
    \set Staff.midiInstrument = #"acoustic grand"
    <<
        \new Voice = "one" \rhUpper
        \new Voice = "two" \rhLower
    >>
}
PianoLH = {
    \clef bass
    \global
    \set Staff.midiInstrument = #"acoustic grand"
    <<
        \new Voice = "one" \lhUpper
        \new Voice = "two" \lhLower
    >>
}

piano = {
    <<
        \set PianoStaff.instrumentName = #"Piano"
        \new Staff = "upper" \PianoRH
        \new Staff = "lower" \PianoLH
    >>
}

% ----- Bass Guitar -----
Bass = \relative c {
    \Key
    c1 | c | c
}
bass = {
    \global
    \set Staff.instrumentName = #"Bass"
    \clef bass
    <<
        \Bass
    >>
}

% ----- Drums -----
up = \drummode {
    \voiceOne
    hh4 <hh sn> hh <hh sn>
    hh4 <hh sn> hh <hh sn>
    hh4 <hh sn> hh <hh sn>
}

```

```

}
down = \drummode {
  \voiceTwo
  bd4 s bd s
  bd4 s bd s
  bd4 s bd s
}

drumContents = {
  \global
  <<
    \set DrumStaff.instrumentName = #"Drums"
    \new DrumVoice \up
    \new DrumVoice \down
  >>
}

%%%%%%%%%% It All Goes Together Here %%%%%%%%%%%%%%

\score {
  <<
    \new StaffGroup = "horns" <<
      \new Staff = "trumpet" \trumpet
      \new Staff = "altosax" \altoSax
      \new ChordNames = "barichords" \bariHarmony
      \new Staff = "barisax" \bariSax
      \new Staff = "trombone" \trombone
    >>

    \new StaffGroup = "rhythm" <<
      \new ChordNames = "chords" \gtrHarmony
      \new Staff = "guitar" \guitar
      \new PianoStaff = "piano" \piano
      \new Staff = "bass" \bass
      \new DrumStaff \drumContents
    >>
  >>
  \layout {
    \context { \Staff \RemoveEmptyStaves }
    \context {
      \Score
      \override BarNumber.padding = #3
      \override RehearsalMark.padding = #2
      skipBars = ##t
    }
  }
  \midi { }
}

```

Song
(tune)

Me

moderato

Swing

Trumpet

Alto Sax

Bari Sax

Trombone

Guitar

Piano

Bass

Drums

B[△] Solo C[#]m⁷

Cm[△] D[△] 9

Annex B GNU Free Documentation License

Version 1.3, 3 November 2008

Copyright © 2000, 2001, 2002, 2007, 2008 Free Software Foundation, Inc.

<http://fsf.org/>

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondly, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of “copyleft”, which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The “Document”, below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as “you”. You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A “Modified Version” of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A “Secondary Section” is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document’s overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The “Invariant Sections” are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The “Cover Texts” are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A “Transparent” copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not “Transparent” is called “Opaque”.

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The “Title Page” means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, “Title Page” means the text near the most prominent appearance of the work’s title, preceding the beginning of the body of the text.

The “publisher” means any person or entity that distributes copies of the Document to the public.

A section “Entitled XYZ” means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as “Acknowledgements”, “Dedications”, “Endorsements”, or “History”.) To “Preserve the Title” of such a section when you modify the Document means that it remains a section “Entitled XYZ” according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document’s license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both

covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

- A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
- B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
- C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
- D. Preserve all the copyright notices of the Document.
- E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
- F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
- G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
- H. Include an unaltered copy of this License.
- I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its

Title Page, then add an item describing the Modified Version as stated in the previous sentence.

- J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the “History” section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
- K. For any section Entitled “Acknowledgements” or “Dedications”, Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
- L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
- M. Delete any section Entitled “Endorsements”. Such a section may not be included in the Modified Version.
- N. Do not retitle any existing section to be Entitled “Endorsements” or to conflict in title with any Invariant Section.
- O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version’s license notice. These titles must be distinct from any other section titles.

You may add a section Entitled “Endorsements”, provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled “History” in the various original documents, forming one section Entitled “History”; likewise combine any sections Entitled “Acknowledgements”, and any sections Entitled “Dedications”. You must delete all sections Entitled “Endorsements.”

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an “aggregate” if the copyright resulting from the compilation is not used to limit the legal rights of the compilation’s users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document’s Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled “Acknowledgements”, “Dedications”, or “History”, the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See <http://www.gnu.org/copyleft/>.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License “or any later version” applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy’s public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

“Massive Multiauthor Collaboration Site” (or “MMC Site”) means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A “Massive Multiauthor Collaboration” (or “MMC”) contained in the site means any set of copyrightable works thus published on the MMC site.

“CC-BY-SA” means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

“Incorporate” means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is “eligible for relicensing” if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.  
Permission is granted to copy, distribute and/or modify this document  
under the terms of the GNU Free Documentation License, Version 1.3  
or any later version published by the Free Software Foundation;  
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover  
Texts.  A copy of the license is included in the section entitled ``GNU  
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the “with...Texts.” line with this:

```
with the Invariant Sections being list their titles, with  
the Front-Cover Texts being list, and with the Back-Cover Texts  
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Annex C Index del LilyPond

!		\<.....	24
!	24	\>.....	24
%		\\.....	31, 48
%.....	16	\absolute.....	39
%{ ... %}.....	16	\acciaccatura.....	26
,		\addlyrics.....	31
'	12	\addlyrics example.....	94
(\addlyrics, example.....	98
(...)	22	\appoggiatura.....	26
,		\autoBeamOff.....	25, 57
,	12	\autoBeamOn.....	25
.		\book.....	41, 42
.	16	\clef.....	16
<		\consists.....	68
<.....	24, 30	\context.....	66
< ... >	30	\f.....	24
<<.....	27, 31	\ff.....	24
<< ... >>.....	27	\grace.....	26
<< ... \\ ... >>.....	31	\header.....	38, 42
<< \\ >>.....	48	\hide.....	102
>		\key.....	21
>.....	24, 30	\layout.....	42, 69
>>.....	27, 31	\lyricmode.....	57
[\lyricsto.....	56
[.....	25	\major.....	21
[...]	25	\markup.....	24
]		\mf.....	24
]	25	\midi.....	42
^		\minor.....	21
^	23	\mp.....	24
-		\new.....	28, 60
-	23	\new ChoirStaff.....	57
\		\new Lyrics.....	56
\!.....	24	\new Staff.....	28
\(... \)	22	\new Voice.....	53
		\omit.....	101, 105, 139
		\omit, example.....	139
		\once.....	90, 96
		\oneVoice.....	53
		\override.....	89
		\overrideProperty.....	91
		\p.....	24
		\partial.....	26
		\pp.....	24
		\relative.....	12
		\remove.....	68
		\revert.....	90, 97
		\score.....	41, 43
		\set.....	63
		\set, example of using.....	113
		\shiftOff.....	56, 125
		\shiftOn.....	56, 125
		\shiftOnn.....	56, 125
		\shiftOnnn.....	56, 125
		\single.....	93
		\startTextSpan.....	114
		\stopTextSpan.....	114
		\tempo.....	15
		\textLengthOff.....	117
		\textLengthOn.....	117
		\time.....	15

<code>\tuplet</code>	26
<code>\tweak</code>	91
<code>\tweak, Accidental</code>	92
<code>\tweak, example</code>	91, 92
<code>\tweak, specific layout object</code>	92
<code>\unset</code>	63
<code>\version</code>	16
<code>\voiceFour</code>	53
<code>\voiceFourStyle</code>	51
<code>\voiceNeutralStyle</code>	51
<code>\voiceOne</code>	53
<code>\voiceOneStyle</code>	51
<code>\voiceThree</code>	53
<code>\voiceThreeStyle</code>	51
<code>\voiceTwo</code>	53
<code>\voiceTwoStyle</code>	51
<code>\with</code>	66
<code>\with, example</code>	105, 106, 108

~

~	22
---------	----

A

absolut, mode.....	38
absoluts, noms de nota	38
absoluts, valors d'altura	38
accent	23
acciacatura.....	26
Accidental, example of overriding	127
AccidentalPlacement, example of overriding	127
accidentals, alteracions	21
acords	30
acords en front a veus.....	48
addició de gravadors	68
addició de text.....	24
<code>addlyrics</code>	31
adjustments, using variables for	141
<code>alignAboveContext</code> property, example ..	105, 106, 108
aligning objects on a baseline	127
alineació de la lletra.....	32
alteracions	21
alteracions i armadures.....	21
alteracions i mode relatiu	12
alteracions y armadura.....	21
alto	16
altures	12
altures, valors absoluts d'	38
àmbit, gravador del	68
ampliabilitat de las pautes	81
anacrusi	26
aniuat d'expressions musicals.....	55
aniuat de construccions simultànies.....	55
aniuat de veus	55
anomenar contextos	61
apòstrof.....	12
appoggiatura.....	26
armadura	21
articulació	23
articulations and slurs.....	116
assignació de variables	36
<code>autoBeamOff</code>	25, 57
<code>autoBeamOn</code>	25

automàtiques, barres	25
----------------------------	----

B

baix	16
bar numbers, tweaking placement	115
BarLine, example of overriding ...	100, 101, 102, 103, 104
barrat	25
barrat i lletra	57
barres automàtiques.....	25
barres de corxera.....	25
barres manuals.....	25
Beam, example of overriding	130
beams, controlling manually	129
bemoll	21
bemoll, doble	21
blanca	14
bloc d'encapçalaments.....	38
bloc, comentari de	16
<code>book</code>	41, 42
book (llibre)	41
book, bloc implícit	42
bound-details property, example	114, 115
bracket, triplet	92
bracket, tuplet	92
break-visibility property.....	102
break-visibility property, example	102

C

cançons	31
capçalera	42
capes	48
caràcters permesos en variables.....	36
center	111
changing size of objects	105
<code>ChoirStaff</code>	29, 57
<code>ChordNames</code>	28
clau.....	16
claudàtors	16
claudàtors i parèntesis, niuat de	47
claudàtors i parèntesis, tancar en front a marcar ..	47
claudàtors i parèntesis, tipus de	47
<code>clef</code>	16
Clef, example of overriding.....	104, 105, 106, 108
clicar sobre els exemples.....	18
col.lisions de notes	56
collisions, notes	125
color property	103
color property, example	90, 92, 103, 104
color property, setting to Scheme procedure	147
color, rgb	104
color, X11	104
columna de notes	56
com llegir el manual.....	18
coma	12
combinar expressions en paral.lel	27
comentari de bloc	17
comentari de línia	16, 17
comentaris	16, 17
cometa simple.....	12
compàs parcial	26
compàs, indicació de	15

compilació	1
composta, expressió musical	27, 43
comuns, errors	18
concorrent, música	48
consells per construir fitxers	18
consists	68
construir fitxers, consells	18
context	28
context	66
context de notació	28
context Voice (de veu)	48
context, finding	96
context, identifying correct	96
context, propietats de	63
context, propietats de, establiment amb \with	66
context, propietats de, modificació	63
contextos de veu, creació de	53
contextos implícits	41
contextos, creació de	60
contextos, establir propietats de, amb \context	66
contextos, explicació dels	59
contextos, nomenament	61
contingut del bloc score	43
contingut en front a presentació	21
controlling tuplets, slurs, phrasing slurs, and beams manually	129
cor, sistema de	29, 57
creació de contextos	60
crescendo	24

D

decrecendo	24
default properties, reverting to	97
depuració d'errors	18
desplaçament, instruccions de	56
digitacions	23
dinàmica, indicacions de	24
direction property, example	92, 111, 112
disposició	42
distances	108
diverses estrofes vocals	58
diverses veus	31, 48
diversos pentagrames	27, 28
diversos pentagrames i lletra	35
do en quarta, clau de	16
do en tercera, clau de	16
doble bemoll	21
doble sostingut	21
down	111
duracions de les notes	14
duracions de notes en acords	30
DynamicLineSpanner, example of overriding	127
dynamics, tweaking placement	118
DynamicText, example of overriding	119, 127

E

editors de text	1
eliminació de gravadors	68
encapçalaments	38
entrada, format de l'	41
errors comuns	18
es	21

escriure una partitura, exemple	79
eses	21
espaiadores, notes	55
espais, insensible a	16
establir propietats en contextos	63
estrofes, diverses, vocals	58
example of \omit	139
execució del LilyPond sota el MacOS X	2
execució del LilyPond sota el Windows	6
execució del LilyPond sota l'Unix	11
exemple d'escriptura d'una partitura	79
exemple inicial	1
exemples, clicar	18
expressió musical	27
expressió musical composta	43
expressions	16
expressions paral·leles	27
extra-offset property	125
extra-offset property, example	128
extra-spacing-width	118
extra-spacing-width property	124
extra-spacing-width property, example	119, 127

F

fa, clau de	16
fermata, implementing in MIDI	139
figura amb puntet	14
fingering example	112, 113
fingering, chords	112
Fingering, example of overriding	112, 128
fingering, placement	112
fingeringOrientations property, example	113
fitxer, estructura del	41
fitxers, consells per construir	18
fixing overlapping notation	126
Flag, example of removing	139
font, mida de la	66
font-series property, example	141
font-shape property, example	98, 141
font-size property, example	91
fontSize (mida de la tipografia)	66
fontSize property, example	108
force-hshift property	125
force-hshift property, example	131, 137
fraseig, lligadura de	22

G

grace	26
gràfics, objectes	81
GrandStaff	29
gravadors	62
gravadors, addició	68
gravadors, eliminació	68
grob	88
grob sizing	118
grobs	81
grobs, moving colliding	123
grobs, positioning	128
grobs, properties of	94
grup de pentagrames	29
grups especials	26
guió	32

guió baix..... 32

H

header..... 38, 42
 himne, estructura de..... 58
 horizontal-shift property..... 125

I

identificadors..... 36, 84
 implícit, bloc book..... 42
 implícits, contextos..... 42
 inicial, exemple..... 1
 interface..... 88, 98
 interface properties..... 98
 Internals Reference..... 94
 Internals Reference manual..... 94
 Internals Reference, example of using..... 94
is..... 21
isis..... 21
 italic, exemple..... 98

K

key..... 21

L

layout..... 42, 69
 layout adjustments, using variables for..... 141
 layout object..... 88
 layout objects, properties of..... 94
 layout, efecte de la situació del bloc..... 43
 length..... 108
 línia de extensió..... 32
 línia, comentari de..... 16
 llegir el manual..... 18
 lletra..... 31
 lletra i barrat..... 57
 lletra i diversos pentagrames..... 35
 lletra, alineació de..... 32
 lletra, creació d'un context de..... 56
 lletra, enllaçar amb una veu..... 56
 lletra, paraules polisíl·labes..... 32
 llibre..... 41
 lligadura d'expressió..... 22
 lligadura d'unió..... 22
 lligadura de fraseig..... 22
 lligadures d'expressió en front a lligadures d'unió.. 23
 lligadures que creuen claudàtors..... 49
lyricmode..... 57
Lyrics..... 28, 56
 Lyrics, creació d'un context..... 56
lyricsto..... 56
 LyricText, exemple of overriding..... 98, 141

M

MacOS X, execució del LilyPond..... 2
 macros..... 36
 magstep..... 108
 magstep function, example of using..... 108
major..... 21

majúscules, sensible a..... 1, 16
 manual, lectura del..... 18
 manually controlling tuplets, slurs, phrasing slurs,
 and beams..... 129
Manuals..... 1
 manuals, barres..... 25
 marcat..... 24
markup..... 24
 markup example..... 110
 markup text, allowing collisions..... 117
 matisos..... 24
 melisma..... 32
 menor..... 21
 metrònom, indicacions..... 15
 metronome mark, tweaking placement..... 115
 MetronomeMark, example of overriding..... 126, 140
 mida de la font..... 66
midi..... 42
minor..... 21
 modificar les propietats de context..... 63
 modificar plantilles..... 71
 moving colliding grobs..... 123
 moving colliding objects..... 123
 moving overlapping objects..... 123
 MultiMeasureRest, example of overriding..... 128
 musical, expressió..... 27
 musical, expressió, composta..... 27, 43

N

naming conventions for objects..... 89
 naming conventions for properties..... 89
 negra..... 14
 neutral..... 111
new..... 28, 60
new Staff..... 28
 notació de las duracions..... 14
 notació dels silencis..... 15
 notació senzilla..... 12
 notació, context de..... 28
 note collisions..... 125
 note column..... 125
 NoteColumn, example of overriding..... 131, 137
 NoteHead, example of overriding..... 90, 91, 104, 147
 notes..... 12
 notes d'adorn..... 26
 notes, collisions de..... 56
 notes, duracions de..... 14
 notes, noms absoluts de..... 38
 notes, noms de..... 38
 notes, spreading out with text..... 117
 nous contextos..... 60
 número de versió..... 16

O

object..... 88
 object collision within a staff..... 128
 object properties..... 88
 object, layout..... 88
 objectes gràfics..... 81
 objects, aligning on a baseline..... 127
 objects, changing size of..... 105
 objects, moving colliding..... 123

objects, naming conventions	89
objects, outside-staff	109
objects, positioning	128
objects, removing	139
objects, size of	105
objects, within-staff	109
once	90, 96
once override	96
oneVoice	53
ossias	46
ottava bracket	114
outside-staff objects	109
outside-staff-priority property, example	116, 117
overlapping notation	126
override	89
override command	89
override example	94
override syntax	89
overrideProperty	91
overrideProperty command	91
overrides, using variables for	141
overriding once only	96

P

padding	124, 126
padding property	124
padding property, example	126
paral·leles, expressions	27
paral·leles, expressions, mode relatiu i	27
paraules polisíl·labes en la lletra	32
partial	26
partitura	41, 43
partitura, exemple d'escriptura	79
partitures, diverses	42
pautes, ampliabilitat	81
PDF, fitxer	1
pentagrama, posicionat del	46
pentagrames temporals	46
pentagrames, diversos	27, 28
pentagrames, grup de	29
phrasing slurs, controlling manually	129
PhrasingSlur, example of overriding	129
piano, sistema de	29
PianoStaff	29
plantilla, modificar	71
plantilles	18
plica amunt	52
plica avall	52
pliques, veus i direcció de les	52
plantilla, escriure la vostra pròpia	79
polifonia	27, 31, 48
polifonia i mode relatiu	51
positioning grobs	128
positioning objects	128
positions property	125
positions property, example	129, 130
presentació en front a contingut	21
properties in interfaces	98
properties of grobs	94
properties of layout objects	94
properties, naming conventions	89
properties, object	88
property types	99

propietats que funcionen en contextos	63
propietats, sub-propietats	81
puntet	14

R

rehearsal marks, tweaking placement	115
relatiu, mode	12
relatiu, mode, alteracions i	12
relatiu, mode, expressions paral·leles i	27
relatiu, mode, música simultània i	27
relatiu, mode, polifonia i	51
relative	12
remove	68
removing objects	139
return a veu única	53
revert	90, 97
revert command	90
rgb colors	104
rgb-color	104
right-padding property	124, 127
right-padding property, example	127
ritmes	14
rodona	14

S

SATB, estructura de	58
score	41, 43
Score	28
score (partitura)	41
score, contingut del bloc	43
Script, example of overriding	126
self-alignment-X property	124
self-alignment-X property, example	128
senzilla, notació	12
separador, silenci	31
set	63
shift commands	125
shiftOff	56, 125
shiftOn	56, 125
shiftOnn	56, 125
shiftOnnn	56, 125
silenci	15
silenci separador	31
simultània, música	48
simultània, música, mode relatiu i	27
sistema de cor	29
sistema de piano	29
size of objects	105
size, changing	108
sizing grobs	118
Slur example of overriding	95
Slur, example of overriding	96, 97
slurs and articulations	116
slurs and outside-staff-priority	116
slurs, controlling manually	129
sol, clau de	16
soprano	16
sostingut	21
sostingut, doble	21
spanner	88
spanners	114
staccato	23

Staff	28
staff line spacing, changing.....	108
staff-padding property.....	124
staff-padding property, example.....	127
staff-position property.....	124
staff-position property, example.....	128, 136
staff-space property, example.....	108
StaffSymbol, example of overriding.....	104, 108
startTextSpan	114
stem length, changing.....	108
Stem, example of overriding.....	104, 111, 137
Stem, example of removing.....	139
stencil property.....	100
stencil property, example....	100, 101, 103, 108, 127, 140
stencil property, use of.....	139
stopTextSpan	114
StringNumber, example of overriding.....	128
sub-propietats.....	81

T

tempo	15
tempo, indicacions de.....	15
temporals, pentagrames.....	46
tenor.....	16
text property, example.....	93, 127
text spanner.....	114
text, addició.....	24
textLengthOff	117
textLengthOn	117
TextScript, example of overriding.....	116, 117
TextSpanner, example of overriding.....	114, 115
thickness.....	108
thickness property, example.....	95, 96, 97
Tie, example of overriding.....	136
time	15
TimeSignature, example of overriding.....	102, 103, 104, 105, 106, 108
títol.....	38
tonalitat, armadura de.....	21
tonalitat, armadura de la, establir.....	21
transparency.....	102
transparent property.....	102
transparent property, example.....	93, 102, 137, 140
tresets.....	26
triplet bracket.....	92
triplets, nested.....	92
tuplet	26
tuplet beams, controlling manually.....	129
tuplet bracket.....	92
tuplet-number function, example.....	93
TupletBracket	92
TupletNumber, example of overriding.....	93
tuplets, nested.....	92
tweak	91
tweak command.....	91

tweak, generated from override.....	93
tweaking bar number placement.....	115
tweaking dynamics placement.....	118
tweaking methods.....	89
tweaking metronome mark placement.....	115
tweaking rehearsal mark placement.....	115
tying notes across voices.....	139

U

únic, polifonia en pentagrama.....	31
Unix, execució del LilyPond.....	11
unset	63
up.....	111
ús de les variables.....	36

V

variables.....	36, 43, 84
variables, caràcters permesos en.....	36
variables, definir.....	36
variables, ús de.....	36
variables, using for overrides.....	141
versió.....	16
versions.....	16
veure la música.....	1
veus en front a acords.....	48
veus i direcció de ls pliques.....	52
veus que creuen claudàtors.....	49
veus temporals.....	55
veus, aniuat de.....	55
veus, diverses.....	48
veus, més, en un sol pentagrama.....	31
veus, nomenament de.....	49
veus, retorn a única.....	53
vocal, estructura d'una partitura.....	57
vocal, partitura, diverses estrofes.....	58
Voice	28
Voice (veu), context de.....	48
voiceFour	53
voiceOne	53
voiceThree	53
voiceTwo	53

W

Windows, execució del LilyPond.....	6
with	66
within-staff objects.....	109

X

X11 colors.....	104
x11-color	104
x11-color function, example of using.....	147
x11-color, example of using.....	104