

L^AT_EX for usicians

Guido Gonzato, PhD

Version 1.0.0
January 6, 2019



Abstract

This guide shows how to create L^AT_EX documents that include several kinds of music elements, from very simple to highly complex. Music features may consist of music symbols, song lyrics, guitar chords diagrams, lead sheets, music excerpts, guitar tablatures, multi-page scores.

Music can be produced by T_EX and L^AT_EX directly using packages, and also by external scorewriters. Major packages and programs are listed and briefly described, providing ready-to-use examples.



Contents

1	Introduction	3
1.1	Conventions	4
1.2	Preliminaries	4
1.3	Adding Packages	5
1.4	Including PDF files	6

2	Music Symbols	8
2.1	Using Packages	8
2.1.1	Package <code>musicography</code>	8
2.1.2	Package <code>leadsheets</code>	9
2.1.3	Package <code>lilyglyphs</code>	9
2.2	Using Music Fonts	10
3	Song Lyrics	12
3.1	Package <code>guitar</code>	12
3.2	Package <code>gtrcrd</code>	13
3.3	Package <code>songs</code>	14
3.4	Package <code>musixguit</code>	15
3.5	Package <code>leadsheets</code>	16
3.6	Package <code>songbook</code>	17
3.7	Program: <code>chordii</code>	18
4	Guitar Chord Diagrams	19
4.1	Package <code>gchords</code>	20
4.2	Package <code>songs</code>	21
4.3	Package <code>guitarchordschemes</code>	21
5	Sheet Music	22
5.1	Packages <code>musixtex</code> , <code>m-tx</code>	23
5.2	Package <code>gregoriotex</code>	28
5.3	Program: <code>LilyPond</code>	29
5.3.1	<code>lilypond-book</code>	30
5.3.2	Package <code>lyluatex</code>	31
5.4	Program: <code>abcm2ps</code>	33
5.5	Program: <code>abc2svg</code>	34
5.6	Program: <code>PMW</code>	35
5.7	Program: <code>MUP</code>	36
5.8	Program: <code>MuseScore</code>	37
5.9	Package <code>abc</code>	37
6	Guitar Tablatures	39
6.1	Program: <code>LilyPond</code>	39
6.2	Program: <code>abc2xml.py</code>	40
6.3	Package <code>guitartabs</code>	41
6.4	Using Guitar Pro Tablatures	43

7 Putting It All Together	43
7.1 Package abc , Revisited	43
7.2 Using make	44
8 The End	47
A List of Packages and Programs	48
B Examples	50
B.1 A Complete abc Example	50
B.2 A Complete Songbook Example	54
B.3 A sample Makefile	60



1 Introduction

[And] there is no such hobby that it cannot
be combined with L^AT_EX.

— Clemens Niederberger

I’m a long-time and enthusiastic L^AT_EX user, and I’m also an amateur musician; I play folk music on wind instruments. Years ago, I used L^AT_EX to typeset my [ABC notation tutorial](#), which I’m still maintaining. Writing that tutorial, I included many music snippets along with the corresponding sources, and I was very satisfied with the result. (I hope that other people are satisfied, too.)

Since then, I’ve become interested in other music notation languages. I’ve come across many great programs and many great packages I wasn’t aware of; I’ve collected many snippets and I’ve taken notes. I’m really impressed; there are plenty of options for the musician who is also a L^AT_EX user. In fact, T_EX can typeset music by itself using extensions, such as the MusiX_T_EX family. Besides, L^AT_EX can easily include music produced by other programs.

So, to write L^AT_EX documents that include music we have to solve two problems: how do we make the music in a suitable format? And how do we combine the music with L^AT_EX?

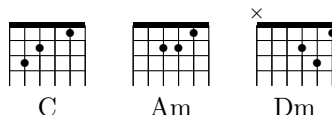
For my own self-training, and hoping to do something useful for other musicians, I have decided to write this manual that explains how to solve these problems. In particular, it shows how to create documents that include many types of music information, from very simple to highly complex:

- **music symbols:** $\sharp \flat$  B G C

- **song lyrics:**

Imagine (John Lennon)
Intro, $\times 2$
C Cmaj7 F
C Imagine there's no Cmaj7 F heaven
C Cmaj7 F It's easy if you try
C Cmaj7 F No hell below us
C Cmaj7 F Above us only sky

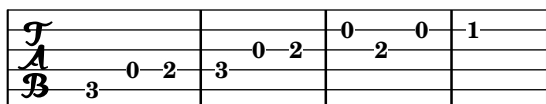
- **guitar chord diagrams:**



- **sheet music:**



- **guitar tablatures:**



This guide aims to provide a roundup of available options, along with ready-to-use examples. We will examine a few $\text{T}_{\text{E}}\text{X}/\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ packages and several programs that produce high-quality musical output; then we will learn how to include them in $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ documents.

1.1 Conventions

This document is meant to be consulted in its PDF version, rather than printed on paper. Links are written in short form whenever possible; for instance, links to packages are written simply as [abc](#) instead of <https://www.ctan.org/pkg/abc>.

Sources ($\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ or notation languages) are shown in a frame with a cyan background; the resulting PDF output, when applicable, has a light background. Command line sessions are shown in a frame, with user commands emphasised in boldface.

All software described in this guide is [free and open source](#).

1.2 Preliminaries

To begin with, we need a working $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$ system. I recommend that you install the latest release of [TeX Live](#), which I used to typeset this manual and all included examples on a GNU/Linux system (TeX Live 2018). On Debian-based GNU/Linux systems, all you need to do is install **texlive-music**; all other needed components will be installed as dependencies. I suppose that

other distributions like [MiKTeX](#) or [MacTeX](#) are just as good, but I have no first-hand experience with them.

Secondly, we need a Bash-based command line environment; this is provided by default in GNU/Linux and macOS systems. I recommend that Windows users install the [MSYS2](#) platform.

In most cases, typesetting is done with the common `pdflatex` command. Some packages, however, require the $\text{X}\text{\LaTeX}$ or $\text{Lua}\text{\LaTeX}$ engines. In that case, you'll also have to install `texlive-xetex` or `texlive-luatex`. In this manual, `pdflatex` is the default command, unless otherwise specified.

This manual is not meant to replace the documentation of the packages and programs that it describes. A few examples are provided to get you started and to whet your appetite, but in many cases they don't cover all available features. The package or program documentation is the primary source of information.

Finally, I assume that you are reasonably familiar with \LaTeX . Should you need some information, fine manuals and tutorials are available at the [\$\text{\LaTeX}\$ info page](#).

1.3 Adding Packages

TeX Live provides a large number of packages, but in the following sections we will deal with packages that may not be included in less recent releases. In this case, you will have to install the missing package manually.

The procedure is simple:

1. create this directory structure:

```
$ mkdir -p $HOME/texmf/tex/latex
```

new packages will be installed in this directory tree.

2. get the package (typically as a zip-compressed directory) from your favourite CTAN mirror; let's call it `foo.zip`

3. unpack it in the right place:

```
$ mkdir $HOME/texmf/tex/latex/foo
```

```
$ mv foo.zip $HOME/texmf/tex/latex/foo
```

```
$ cd $HOME/texmf/tex/latex/foo ; unzip foo.zip
```

4. if no `.sty` file exists, run the command `latex foo.ins` or `latex foo.dtx` to create it;

5. run the command `texhash $HOME/texmf`

Package FOO is now accessible by L^AT_EX.

1.4 Including PDF files

It's reasonable to assume that PDF is the most sensible format for final output. PostScript, SVG, PNG and other formats will not be considered, but it's easy to convert PDF files to these formats using applications like [Inkscape](#) or [ImageMagick](#).

Unless you create music directly using MUSI_XT_EX and related packages (Section 5.1), your L^AT_EX document will include music as PDF files. Such files can be short excerpts, i.e. smaller than a page, or span several pages. These files will be included with the `\includegraphics` command (package [graphicx](#)) or with the `\includepdf` command (package [pdfpages](#)), respectively:

```
\documentclass[oneside]{article}
\usepackage{graphicx}
\usepackage{pdfpages}
\usepackage[a4paper,margin=1.5cm]{geometry}

\begin{document}

This is a short excerpt:

\includegraphics[width=0.8\textwidth]{sample.pdf}

Let's now include several pages:

% pages=- means "all pages"
\includepdf[pages=-,pagecommand={},width=\textwidth]{music.pdf}

\end{document}
```

In the case of short excerpts, we need some means of cropping the PDF to its actual contents (bounding box); PDF files, in fact, are usually created as whole pages. Cropping the PDF is accomplished with the free program [pdftcrop](#), a Perl script that depends on [Ghostscript](#) and [PDFedit](#).

Given a file called `music.pdf`, we crop it with these commands:

```
$ pdftcrop music.pdf
PDFCROP 1.38, 2012/11/02 - Copyright (c) 2002-2012 by Heiko Oberdiek.
```



```
==> 1 page written on 'music-crop.pdf'.  
$ mv -f music-crop.pdf music.pdf  
$ _
```

Pdftocrop doesn't work on MSYS2. However, if you install the official release of [Ghostscript](#) for Windows, you can use the following shell script, `pdfcrop.sh`. It only works with single-page PDF files:

```
#!/bin/sh  
  
# pdfcrop.sh - for MSYS2 and GhostScript  
# Guido Gonzato, PhD. GPL 2 or later.  
  
MYSELF=$(basename $0)  
  
if [ $# -eq 0 ] ; then  
    printf "Usage: ${MYSELF} <file.pdf>\n"  
    printf "This script uses 'gs' to crop a one-page pdf file.\n\n"  
    exit 1  
fi  
  
# GhostScript is installed in C:\Gs  
GS=/c/gs/gs9.26/bin/gswin64c.exe  
INPUT=$1  
PDF=$(basename $1 .pdf)  
OUTPUT=$PDF-cropped.pdf  
GSOPTS="-q -sDEVICE=bbox -dBATCh -dNOPAUSE"  
  
# find out the bounding box  
$GS $GSOPTS $INPUT 2>&1 | grep "%B" > $PDF.bbox  
  
# read bbox coordinates in variables  
read tmp X1 Y1 X2 Y2 < $PDF.bbox  
  
# write the output, cropped to bbox  
$GS -q -o $OUTPUT \  
    -sDEVICE=pdfwrite \  
    -c "[ /CropBox [$X1 $Y1 $X2 $Y2] /PAGES pdfmark" \  
    -f $INPUT  
  
/bin/rm -f $PDF.bbox  
  
echo "$INPUT cropped to $OUTPUT"
```




2 Music Symbols

The simplest music elements we may want to include in our documents are music symbols (*glyphs*). Standard L^AT_EX only provides the math mode commands `\sharp`, `\flat`, and `\natural`: \sharp \flat \natural . Additional glyphs are provided by several packages; moreover, glyphs provided by music fonts are accessible via X_YL^AT_EX and LuaL^AT_EX.

2.1 Using Packages

The impressive [Comprehensive LaTeX Symbol List](#), Section 7, lists packages that provide a handful of music symbols: `textcomp`, `mnsymbol`, `fdsymbol`, `boisik`, `wasysym`, `stix`, and `arev`.

More symbols are provided by `musicography`, `leadsheets`, and `harmony`. These packages work with `pdflatex`, but still provide a fairly limited number of glyphs. Besides, there could be incompatibility between packages because of commands defined using the same name.

2.1.1 Package `musicography`

This package may be missing in less recent releases of TeX Live, so you might have to install it manually. If you only need a limited number of glyphs, then `musicography` may do the job:

```
\documentclass{article}
\usepackage{musicography}
\thispagestyle{empty}
```

```
\begin{document}
```

Musicography makes the music symbol font provided by MusiX_{TeX} available as text font and then uses it to define a number of symbols:

```
\musFlat\ \musSharp\ \musNatural\ \musDoubleFlat\ \musDoubleSharp\
\musWhole\ \musHalf\ \musQuarter\ \musEighth\ \musSixteenth\
\musWholeDotted\ \musHalfDotted\ \musQuarterDotted\
\musEighthDotted\ \musSixteenthDotted\
\meterC\ \meterCutC\ \meterCThree\ \meterCThreeTwo\ \meterCZ\
\musMeter{7}{8}
```


\end{document}

Musicography makes the music symbol font provided by MusiX_{TEX} available as text font and then uses it to define a number of symbols:

♭ # ♯ ♭ × ○ ♪ ♫ ♫ ♫ ○. ♪. ♪. ♪. ♪. **C C C3 C3 CZ 7**

2.1.2 Package **leadsheets**

If you only need a limited number of glyphs, then also **leadsheets** may do the job:

```
\documentclass{article}
\usepackage[minimal]{leadshsheets}
\useleadshsheetslibraries{musicsymbols}
\thispagestyle{empty}
```


```
\begin{document}
```

Leadsheets makes the music symbol font provided by MusiX\TeX\ available as text font and then uses it to define a number of symbols:

\sharp\ \doublesharp\ \flat\ \doubleflat\ \natural\
\trebleclef\ \altoclef\ \bassclef\ \meterC\ \allabreve\
\meter{12}{8}\ \wholorest\ \halfrest\ \quarterrest\
\eighthrest\ \sixteenthrest\ \Break\ \normalbar\
\leftrepeat\ \rightrepeat\ \leftrightrepeat\ \doublebar\
\stopbar

\end{document}

Leadsheets makes the music symbol font provided by MusiX_{TEX} available as text font and then uses it to define a number of symbols:

x b bb z 

2.1.3 Package `lilyglyphs`

By far, the most complete source of music glyphs is the [lilyglyphs](#) package. It provides all the symbols available in the [Emmentaler](#) music font, which

is used by the [LilyPond](#) scorewriter (Section 5.3). LILYGLYPHS only works with X_YL^AT_EX and Lua^AT_EX, and is incompatible with LEADSHEETS and MUSI^XT_EX. Available glyphs and corresponding commands are listed in Section 3 of the package documentation.

This is a very small example of what LILYGLYPHS provides:




```
\documentclass{article}
\usepackage{fontspec}
\usepackage{lilyglyphs}
\thispagestyle{empty} % no page number




















\begin{document}

Lilyglyphs makes the music symbol font provided by LilyPond available
as text font and then uses it to define a number of symbols, some
\clefGInline\ of which \clefCInline\ can be used \clefFInline\ inline:

clefs:~ \clefG\ \clefC\ \clefF\ ~time signatures:~
\lilyTimeC\ \lilyTimeCHalf\ \lilyTimeSignature{7}{8}\
~accidentals:~ \sharp\ \flat\ \natural\ \doublesharp\ \flatflat\
~rests:~ \wholeNoteRest\ \halfNoteRest\ \crotchetRest\
~notes:~ \wholeNote\ \halfNote\ \halfNoteDown\ \quarterNote\
\quarterNoteDotted\
~and much, much more.

\end{document}
```

Lilyglyphs makes the music symbol font provided by LilyPond available as text font and then uses it to define a number of symbols, some  of which  can be used  inline:

clefs:    time signatures:    accidentals:      rests:   
notes:      and much, much more.

2.2 Using Music Fonts

The X_YL^AT_EX and Lua^AT_EX engines use Unicode input by default and support OTF/TTF fonts. We can use these engines to print any character provided by locally installed fonts; these can be listed with the `fc-list` command, provided by the [Fontconfig](#) software.

Some fonts are especially useful for music. Specifically, [Bravura](#) is a free, SMuFL-compliant music font that provides thousands of high-quality

music glyphs. Bravura is available in OpenType format as **Bravura.otf** and **BravuraText.otf**; the first is used for drawing music symbols in scores, the second for inserting music symbols in text.

Each glyph is mapped to a numerical code called *code point*; a comprehensive list of glyphs and the corresponding code points is available at <https://www.smubl.org/version>.

The following source shows how to use the glyphs provided by Bravura. We can define new commands for commonly used glyphs, or directly use the `\char"XXXX` syntax to print Unicode characters specifying their code point. The fonts are expected to be installed in `/usr/share/fonts/`; if you install the fonts in a different directory, you'll also have change the following source accordingly:

```
\documentclass{article}
\usepackage{fontspec}
\thispagestyle{empty}
\newfontfamily\brtxt{BravuraText.otf}[Path=/usr/share/fonts/]
\newfontfamily\brv{Bravura.otf}[Path=/usr/share/fonts/]

\newcommand{\clefGi}      {{\brtxt \char"E050}}
\newcommand{\clefCi}      {{\brtxt \char"E05C}}
\newcommand{\clefFi}      {{\brtxt \char"E062}}











\newcommand{\clefG}       {{\brv \char"E050}}
\newcommand{\clefC}       {{\brv \char"E05C}}
\newcommand{\clefF}       {{\brv \char"E062}}
\newcommand{\timeC}       {{\brv \char"E08A}}
\newcommand{\timeCHalf}   {{\brv \char"E08B}}
\renewcommand{\flat}      {{\brv \char"E260}}
\renewcommand{\natural}   {{\brv \char"E261}}
\renewcommand{\sharp}     {{\brv \char"E262}}
\newcommand{\wholeNote}   {{\brv \char"E1D2}}
\newcommand{\halfNote}    {{\brv \char"E1D3}}
\newcommand{\halfNoteDown}{{\brv \char"E1D4}}
\newcommand{\quarterNote} {{\brv \char"E1D5}}

\begin{document}
```

The Bravura and BravuraText Music fonts provide thousands of music symbols. BravuraText glyphs are specifically `\clefGi` designed `\clefFi` to be used `\clefCi` inline:

```
clefs:~ \clefG\ \clefC\ \clefF\ ~time signatures:~
\timeC\ \timeCHalf\ ~accidentals:~ \sharp\ \flat\ \natural\
~notes:~ \wholeNote\ \halfNote\ \halfNoteDown\ \quarterNote\
```


\end{document}

clefs:    time signatures:   accidentals:   notes:    and much, much more.



3.1 Package `guitar`

```
\documentclass{article}
\thispagestyle{empty}
\usepackage{guitar}

\begin{document}

\begin{guitar}
  \textbf{Imagine (John Lennon)}

  \emph{Intro, $\times$ 2}
  % use tilde to add some space
  [C] ~ [Cmaj7] ~ [F] ~
  % if chords overlap, add | at the end of the chord, like:
  [C]Imagine there's [Cmaj7|]{no} [F]heaven
  [C]It's easy if [Cmaj7|]{you} [F]try
```



```

[C]No hell [Cmaj7|]{below} [F]us
[C]Above us [Cmaj7|]{only} [F]sky
\end{guitar}

\end{document}

```

Imagine (John Lennon)

```

Intro, × 2
C Cmaj7 F

C Cmaj7 F
Imagine there's no heaven
C Cmaj7 F
It's easy if you try
C Cmaj7 F
No hell below us
C Cmaj7 F
Above us only sky

```

3.2 Package **gtrcrd**

This is another basic and simple to use package, with some customisation options. Chords overlaps need manual adjustment.

```

\documentclass{article}
\thispagestyle{empty}
\usepackage{gtrcrd}

\setlength{\crdheight}{2ex} % reduce spacing
\def\crdfont{\footnotesize \itshape \sffamily} % chord font
\setlength{\parindent}{0pt}

\begin{document}

\textbf{Imagine (John Lennon)}

\emph{Intro, $\times$ 2}

% if chords overlap, use \hspace:
\C {\hspace{3mm}} \C[maj7] {\hspace{10mm}} \F ~

\C Imagine there's \C[maj7] no {\hspace{5mm}} \F heaven

\C It's easy if \C[maj7] you {\hspace{2mm}} \F try

```



```
\C No hell \C[maj7] below \F us

\C Above us \C[maj7] only {\hspace{2mm}} \F sky
\end{document}
```

Imagine (John Lennon)
Intro, × 2
C Cmaj7 F

C Cmaj7 F
 Imagine there's no heaven
C Cmaj7 F
 It's easy if you try
C Cmaj7 F
 No hell below us
C Cmaj7 F
 Above us only sky

3.3 Package *songs*

This is a very powerful package that provides many features: chords above lyrics, guitar chords diagrams, transposition, index generation, multiple columns, and more. It allows for the creation of complete songbooks; its guitar chord diagrams capabilities will be explained in Section 4.2.

```
\documentclass{article}
\thispagestyle{empty}
\usepackage[chorded]{songs}

\begin{document}

\renewcommand{\stitlefont}{\rm\large}
\renewcommand{\lyricfont}{\small}
\renewcommand{\printchord}{\it\small}

\begin{songs}{}
\beginsong{Imagine}[by={John Lennon}]
\beginverse
  \emph{Intro, $\times$ 2}
  \[C] \[Cmaj7] \[F]
  \[C]Imagine there's \[Cmaj7]no \[F]heaven
  \[C]It's easy if \[Cmaj7]you \[F]try
  \[C]No hell \[Cmaj7]below \[F]us
  \[C]Above us \[Cmaj7]only \[F]sky
\endverse
\end{songs}
```



```
\endsong
\end{songs}

\end{document}
```

3.4 Package *musixguit*

This package is integrated with *musixtex* (Section 5.1), and is also capable of producing sheet music and guitar chord diagrams. Its documentation is written in German; if you can't read it don't worry, the provided examples are easy to understand. Chords overlaps need manual adjustment. A minimal example:

```
\documentclass{article}
\usepackage{musixguit}
\thispagestyle{empty}

\begin{document}

\textbf{Imagine (John Lennon)}

\begin{song}

\emph{Intro, $\times$ 2}

\chord{C} ~ \chord{Cmaj7} {\hspace{8mm}} \chord{F}
```



```

\chord{C}Imagine there's \chord{Cmaj7}no~~~ \chord{F}heaven
\chord{C}It's easy if \chord{Cmaj7}you~~ \chord{F}try
\chord{C}No hell \chord{Cmaj7}below \chord{F}us
\chord{C}Above us \chord{Cmaj7}only~ \chord{F}sky
\end{song}
\end{document}

```

Imagine (John Lennon)

Intro, × 2

C Cmaj7 F

```

C           Cmaj7F
Imagine there's no   heaven
C           Cmaj7 F
It's easy if you   try
C           Cmaj7 F
No hell below us
C           Cmaj7 F
Above us only   sky

```

3.5 Package **leadsheets**

This package provides many features: music symbols, chords, MuseJazz style, a **song** and a **verse** environment, transposition, a **leadsheet** class, and templates. Overall, this package allows for the creation of complete songbooks. A minimal example:

```

\documentclass{article}
\usepackage[full]{leadsheets}
\thispagestyle{empty}

\begin{document}

\begin{song}{title={Imagine}, music={John Lennon}}
\begin{verse}

\chord{C}Imagine there's \chord{Cmaj7}no \chord{F}heaven

```



```

\chord{C}It's easy if \chord{Cmaj7}you \chord{F}try
\chord{C}No hell \chord{Cmaj7}below \chord{F}us
\chord{C}Above us \chord{Cmaj7}only \chord{F}sky
\end{verse}
\end{song}
\end{document}

```

Imagine

```

      C          Cmaj7  F
Imagine there's no heaven
      C          Cmaj7  F
It's easy if you try
      C          Cmaj7  F
No hell below us
      C          Cmaj7  F
Above us only sky

```

3.6 Package *songbook*

This is another powerful package that provides support for chords, songs, overhead transparencies, and index generation. A minimal example:

```

\documentclass{article}
\usepackage[chordbk]{songbook}
\thispagestyle{empty}

\begin{document}

\textbf{Imagine (John Lennon)}

% \medskip

\emph{Intro, $\times$ 2}

\Ch{C}~ \Ch{Cmaj7}~ \Ch{F}~

\Ch{C}{Imagine} there's \Ch{Cmaj7}{no} \Ch{F}heaven

```



```

\Ch{C}{It's} easy if \Ch{Cmaj7}{you} \Ch{F}{try}

\Ch{C}No hell \Ch{Cmaj7}{below} \Ch{F}us

\Ch{C}{Above} us \Ch{Cmaj7}{only} \Ch{F}{sky}

\end{document}

```

Imagine (John Lennon)

Intro, × 2

C Cmaj7 F

C Cmaj7 F
Imagine there's no heaven

C Cmaj7 F
It's easy if you try

C Cmaj7 F
No hell below us

C Cmaj7 F
Above us only sky

3.7 Program: *chordii*

Chordii is a free command-line program, released under the [GNU GPL](#). It uses a simple text notation to typeset songs in PostScript format, complete with chords and guitar chord grid.

This is the source of a song written in Chordii format:

```

{titles:left}
{title:Imagine}
{st:John Lennon}

(Intro, x 2)
[C] [Cmaj7] [F]
[C]Imagine there's [Cmaj7]no [F]heaven
[C]It's easy if [Cmaj7]you [F]try
[C]No hell [Cmaj7]below [F]us
[C]Above us [Cmaj7]only [F]sky

```

We typeset the score with these commands:


```
$ chordii -a imagine.cho > imagine.ps
$ ps2pdf imagine.ps
$ _
```

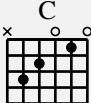
The `-a` switch means “Automatic single space lines without chords”. The resulting song is:

Imagine
John Lennon

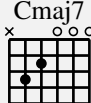
(Intro, x 2)
CCmaj7 F

C *Cmaj7 F*
Imagine there's no heaven
C *Cmaj7 F*
It's easy if you try
C *Cmaj7 F*
No hell below us
C *Cmaj7 F*
Above us only sky

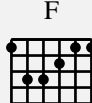
C



Cmaj7



F



The chord grid at the bottom is automatically added by the program.



4 Guitar Chord Diagrams

Guitar players may need to print *guitar chords diagrams* and *guitar tablatures*; both can be made with L^AT_EX packages and external programs. As a matter of fact, these musical features are not limited to the guitar; chord diagrams and tablatures apply to other stringed instruments as well.

Tablatures are a form of musical notation that is usually employed to show how a *melody* should be fingered on the fretboard. Since tablatures are just a special form of music notation, we will deal with them after the section about sheet music. For the moment, let’s see how to do guitar chord diagrams.

4.1 Package **gchords**

This package makes it possible to print guitar chord diagrams using the `\chord` command that employs a simple syntax:

`\chord{fret number}{fingering}{chord name}`

- *fret number* can be `{t}`, which means top fret, or `{t}` followed by a digit that denotes the fret;
- *fingering* is explained in the example below;
- *chord name* is the given chord name.

The `\chords` command prints a row of chords, each defined by a `\chord` command:

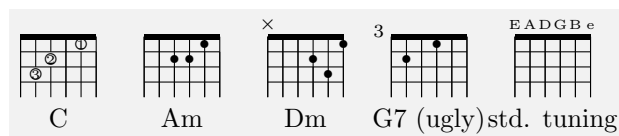
```
\documentclass{article}
\usepackage{gchords}
\thispagestyle{empty}

\begin{document}

\def\numfrets{4}

\chords{ % print a row of chords
% fingers: n, x, o, p{n}, f{n}
% C chord: finger 3 pos. 3, finger 2 pos. 2, finger 1 pos. 1
\chord{t}{n,f3p3,f2p2,n,f1p1,n}{C}
\chord{t}{n,n,p2,p2,p1,n}{Am}
\chord{t}{x,n,n,p2,p3,p1}{Dm}
\chord{t3}{n,p2,n,p1,n,n}{G7 (ugly)}
{\tiny % font size for string labels, t{X}
\chord{t}{t{E}n,t{A}n,t{D}n,t{G}n,t{B}n,t{e}n}
{std. tuning} }
}

\end{document}
```



4.2 Package `songs`

We met this package in Section 3.3. It provides an easy way to print guitar chord diagrams (referred to as “guitar tablatures” in the package). The `\gtab` command defines chord diagrams using a very simple syntax:

```
\gtab{chord name} {fret:strings:fingering}
```

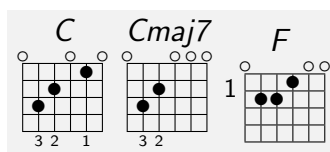
- *chord name* is the given chord name;
- *fret* is an optional fret number;
- *strings* is a string of digits denoting the strings that compose the chord;
- *fingering* is an optional string of digits denoting the fingers to use.

```
\documentclass{article}
\thispagestyle{empty}
\usepackage[chorded]{songs}

\begin{document}

% \gtab{chord name}{fret:strings:fingering}
\gtab{C}{032010:032010} \gtab{Cmaj7}{032000:032000}
\gtab{F}{1:022100}

\end{document}
```



4.3 Package `guitarchordschemes`

This package enables the creation of large guitar chord diagrams. The main command is `\chordscheme`, and is quite self-explanatory:

```
\documentclass{article}
\usepackage{guitarchordschemes}
\thispagestyle{empty}
```



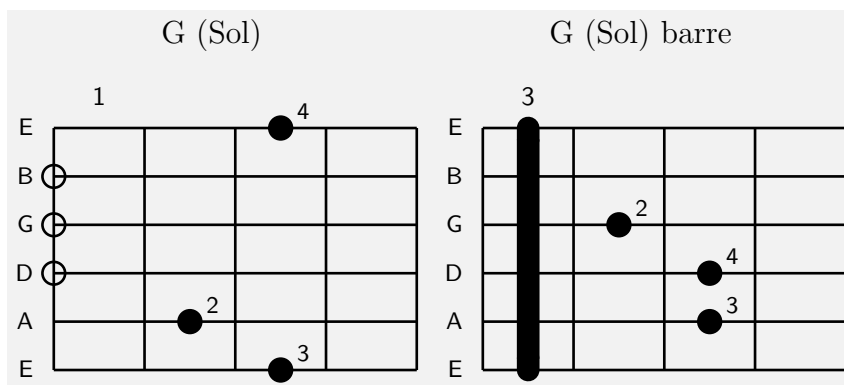
```

\begin{document}

\chordscheme[
name = G (Sol),      % chord name
position = 1,        % first fret position
finger = {2/5:2} ,   % fret, string, finger
finger = {3/6:3} ,   % fret, string, finger
finger = {3/1:4} ,   % fret, string, finger
ring = {2,3,4}       % open strings
]
%
\chordscheme[
name = G (Sol) barre,
position = 3,
barre = 1/1-6,       % fret, string range
finger = {2/3:2},
finger = {3/5:3},
finger = {3/4:4},
]

\end{document}

```



5 Sheet Music

In this section, we will learn how to produce sheet music in PDF format for inclusion in \LaTeX documents. Sheet music can be produced by \TeX itself, but we can use the output produced by one of the several free and open source scorewriters.

In the next few sections, we will briefly examine each option. Musical notation languages will not be explained; sources are listed to give practical examples and to assess the language complexity.

5.1 Packages *musixtex*, *m-tx*

In the beginning, \TeX could not typeset music, and everyone was unhappy. Then the *mtex* (aka \MuTeX) package was created, but it was limited and not very easy to use. Then \MuTeX begat \MusicTeX , which begat *pmtex*; then \MusicTeX begat *musixtex*, which begat *pmx*, which begat *m-tx*. I hope I got the storyline right. Each package was a simpler interface to its predecessors.

In reverse order: *M-Tx* is a preprocessor to *PMX*, which in turn is a preprocessor for \MusixTeX , which does the actual music typesetting via \TeX . It goes without saying that *M-Tx* is the simplest to use, while \MusixTeX is the most difficult to use but also the most powerful. Other packages related to \MusixTeX are the following:

- *autosp* generates note-spacing commands for \MusixTeX scores;
- *bagpipe* provides support for typesetting bagpipe music;
- *bizantinemusic* facilitates the writing of Byzantine music;
- *figbas* provides mini fonts for figured bass notation;
- *gregoriotex* provides engraving of Gregorian Chant (Section 5.2);
- *lyluatex* provides commands to include LilyPond scores in \Lua\TeX documents (Section 5.3.2);
- *musixtnt* is an extension library that enables transformations of the effect of notes commands;
- *pmxchords* produces chord information to go with *pmx* output;
- *snote* provides shape notes for \MusixTeX ;
- *texmuse* is a music typesetting system using \TeX and Metafont.

\MusixTeX is quite low-level, and the user must take care of such details as beam slope and note spacing; several examples are available [here](#). \MusixTeX input can also be embedded in \LaTeX documents:


```

\documentclass{article}
\usepackage{musixtex}
\thispagestyle{empty}

\begin{document}

A short music excerpt in MusiX\TeX:

\medskip

\begin{music}
  \smallmusicsize
  \instrumentnumber{1}
  \setstaves1{1}
  \generalmeter{\meterC}
  \nobarnumbers
  \startextract
  % bar 1
  \Notes \qu c \en % C
  \notes \ibu1d2\qb1d\tbu1\qb1e \en % beamed DE
  \notes \ibu1g2\qb1f\qb1g%
    \qb1{'a}\tbu1\qb1b \en % beamed FGAB
  \bar % bar 2
  \Notes \ql{'c} \en % c
  \notes \ibu1{'b'}{-3}%
    \qb1b\tbu1\qb1a \en % beamed BA
  \notes \ibu1{g}{-3}%
    \qb1g\qb1f\qb1e\tbu1\qb1d \en % beamed GFED
  \bar % bar 3
  \notes \ibu1f0\qb1c\qb1g\qb1e\tbu1\qb1g% % beamed CGEG
    \ibu1f0\qb1c\qb1g\qb1e\tbu1\qb1g \en % beamed CGEG
  \bar % bar 4
  \Notes \qu c\qu e\qu c\qp \en % CEC
  \endextract
\end{music}

End of the excerpt.
\end{document}

```


A short music excerpt in MusiX_{TEX}:



End of the excerpt.

It looks a bit arcane, and I'll point you to the [documentation](#) for explanations.

M-Tx employs a much simpler notation than MusiX_{TEX}. This is a standalone music sample (`sample-mtx.mtx`) that produces the same music as the above MusiX_{TEX} score:

```
% music sample in M-Tx notation
```

```
Title: Music sample in M-tx
```

```
Style: Solo
```

```
Meter: C
```

```
Width: 140mm
```

```
c4 d8 e f g a b | c4 b8 a g f e d | c8 g+ e g c- g+ e g | c4- e c r |
```

We typeset the score with this command:

```
$ musixtex sample-mtx.mtx
```

```
This is musixtex.lua version 0.17a.
```

```
==> This is M-Tx 0.62 (Music from TeXt) <08 February 2016>
```

```
==>> Input from file sample-mtx.mtx
```

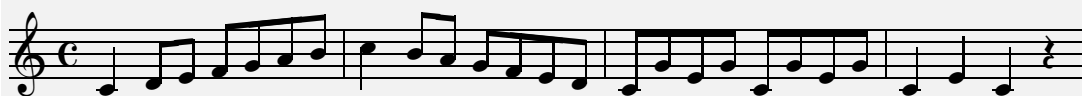
```
...
```

```
sample-mtx.pdf generated by ps2pdf.
```

```
$ _
```

which produces `sample-mtx.pdf`. In older versions of M-Tx, the command was `m-tx`, which has been retired and replaced by `musixtex`. This is the resulting score:

Music sample in M-Tx



Music in M-Tx format can be easily included in L^AT_EX documents. M-Tx provides the **Score**, **excerpts**, and **mus** environments to include complete pieces, short excerpts, and inline short excerpts respectively. Let's see how to use **excerpts** and **mus**; a few steps are required.

First of all, the L^AT_EX source that includes the M-Tx music files must not have a **.tex** extension; **.ltx** or **.latex** are ok. Let's call our sample file **sample-latexmtx.ltx**.

Secondly, we need the **mtxlatex.sty** style file, which is not installed by default but is found in the M-Tx documentation directory. In TeX Live, **mtxlatex.sty** is located in directory **/usr/share/doc/texlive-doc/generic/m-tx**. This file must be copied to the same directory as the source file; all included M-Tx files must be copied there too.

We already met **sample-mtx.mtx**; the following is another short excerpt called **scale.mtx**. It produces a 30mm wide scale:

```
Style: Solo
Meter: C
Size: 13pt
Width: 30mm

c8 d e f g2 |
```

Finally, this is the main file **sample-latexmtx.ltx**:

```
\documentclass[12pt]{article}
\usepackage{mtxlatex} % usually not installed
\thispagestyle{empty}

\mtxlatex

\begin{document}

This \LaTeX{} document includes music written in M-Tx. The \texttt{mus}
environment includes music inline: \begin{mus} \input{scale.tex}
\end{mus} , while the \texttt{excerpts} environment is used for longer
excerpts:

\medskip

\begin{excerpts}
  \input{sample-mtx.tex}
```



```
\end{excerpts}

\medskip

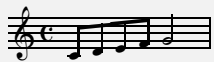
End of document.

\end{document}
```

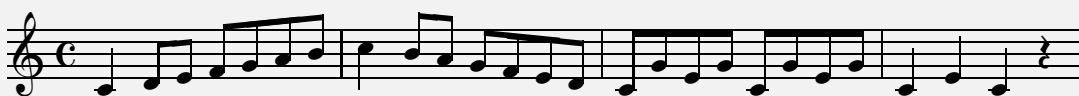
We typeset the document with this command:

```
$ musixtex sample-mtx scale sample-latexmtx.ltx
...
sample-latexmtx.pdf generated by ps2pdf.
$ _
```

Please note that we omitted the M-Tx file extensions, and that the main file is the last in the command line. We obtain this output:

This L^AT_EX document includes music written in M-Tx. The `mus` environment includes music inline: , while the `excerpts` environment is used for longer excerpts:

Music sample in M-Tx



End of document.

Please also note that the M-Tx excerpts cannot be resized as if they were included PDF files: their geometry is set in the M-Tx source!

5.2 Package *gregoriotex*

This package typesets Gregorian chant. Input is in GABC format, a simple text notation inspired by the [ABC notation](#) that we'll examine later on. Conversion requires Lua \LaTeX .

This is a sample GABC source, **kyrie.gabc**, taken from the package documentation:

```
name:Kyrie XVII;
%%
(c4)KY(f)ri(gfg)e(h.) *()
e(ixjvIH'GhvF'E)lé(ghg')i(g)son.(f.)
<i>bis</i>(::)
```

We also need a \LaTeX source, **kyrie.tex**:

```
\documentclass{article}
\thispagestyle{empty}
\usepackage[autocompile]{gregoriotex}

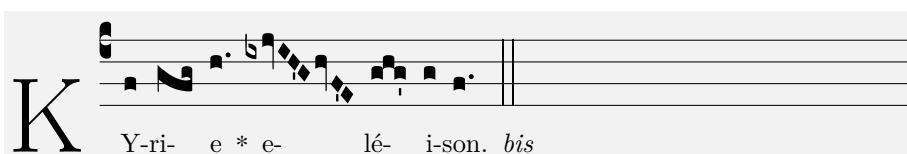
\begin{document}

\gregorioscore{kyrie}

\end{document}
```

We typeset the score with this command:

```
$ lualatex kyrie.tex
...
Output written on kyrie.pdf (1 page, 11290 bytes).
Transcript written on kyrie.log.
$ _
```



5.3 Program: *LilyPond*

LilyPond is a free and multiplatform scorewriter, released under the GNU GPL. From the LilyPond home page:

LilyPond is a music engraving program, devoted to producing the highest-quality sheet music possible. It brings the aesthetics of traditionally engraved music to computer printouts. LilyPond is free software and part of the [GNU Project](#).

LilyPond uses a simple text notation for music input; output is PDF by default. This is our usual music sample in LilyPond format, **sample-lily-pond.ly**:

```
% music sample in LilyPond notation

\version "2.18.2"

\paper{
  indent = 0 \mm
}

\header {
  title = "Music sample in LilyPond"
  tagline = "" % no footer
}

\relative c' {
  \time 4/4
  \clef treble
  c4 d8 e f8 g a b | c4 b8 a g8 f e d |
  c8 g' e g c,8 g' e g | c,4 e c r \bar "|."
}
```

We typeset the document with this command:

```
$ lilypond sample-lilypond.ly
GNU LilyPond 2.18.2
Processing 'sample-lilypond.ly'
Parsing...
Interpreting music...
Preprocessing graphical objects...
```



```

Finding the ideal number of pages...
Fitting music on 1 page...
Drawing systems...
Layout output to 'sample-lilypond.ps'...
Converting to './sample-lilypond.pdf'...
Success: compilation successfully completed
$ _

```

LilyPond is capable of typesetting many different kinds of music, and it is among the most complete and powerful scorewriters available. If you want to find out more, the documentation page is [here](#).

5.3.1 lilypond-book

LilyPond also provides the **lilypond-book** command that can be used to easily embed LilyPond sources in special \LaTeX documents. These should have a **.lytex** extension, like the following source (**sample-lilybook.lytex**):

```

\documentclass{article}
\thispagestyle{empty}

\begin{document}

This is a LilyPond snippet \begin{lilypond} {c' e' g'}
\end{lilypond} embedded in the \LaTeX{} source.

This is another LilyPond excerpt that uses the \texttt{lilypond}
environment:

\medskip

\begin{lilypond}
  \score {
    <<
    \relative c' {
      \time 4/4
      \clef treble
      c4 d8 e f8 g a b | c4 b8 a g8 f e d |
      c8 g' e g c,8 g' e g | c,4 e c r \bar "|."
    }
    >>
  } % end of score
\end{lilypond}

```




```
\medskip

End of document.


\end{document}
```

We typeset the score with these commands:

```
$ lilypond-book -f latex -o /tmp \
  --lily-output-dir=/tmp sample-lilybook.lytex
lilypond-book (GNU LilyPond) 2.18.2
...
Writing '/tmp/sample-lilybook.tex'...
$ cd /tmp
$ pdflatex sample-lilybook.tex
...
Output written on sample-lilybook.pdf (1 page, 60934 bytes).
Transcript written on sample-lilybook.log.
$ _
```

This is a LilyPond snippet  embedded in the \LaTeX source.

This is another LilyPond excerpt that uses the `lilypond` environment:



End of document.

5.3.2 Package `lyluatex`

This package may be missing in less recent releases of TeX Live, so you might have to install it manually. `LYLUATEX` provides a native \LaTeX environ-

ment that offers the same functionality as `lilypond-book`, even though the output is different. This source is `sample-lyluatex.tex`:

```
\documentclass{article}
\usepackage{lyluatex}
\thispagestyle{empty}

\begin{document}

This is a LilyPond snippet \lilypond{c' e' g'} embedded in the text
using the \verb|\lilypond| command.

This is another LilyPond excerpt that uses the \texttt{lilypond}
environment:

\medskip

\begin{lilypond}
  \score {
    <<
    \relative c' {
      \time 4/4
      \clef treble
      c4 d8 e f8 g a b | c4 b8 a g8 f e d |
      c8 g' e g c,8 g' e g | c,4 e c r \bar "|."
    }
    >>
  } % end of score
\end{lilypond}

\medskip

End of document.

\end{document}
```

We typeset the score with this command:

```
$ lualatex --shell-escape sample-lyluatex.tex
...
Output written on sample-lyluatex.pdf (1 page, 22334 bytes).
Transcript written on sample-lyluatex.log.
$ _
```


This is a LilyPond snippet  embedded in the text using the `\lilypond` command.

This is another LilyPond excerpt that uses the `lilypond` environment:



End of document.

5.4 Program: *abcm2ps*

abcm2ps is a free and multiplatform scorewriter, released under the GNU GPL.

This program is currently one of the best implementations of the [ABC notation](#), which describes itself as:

*the text-based music notation system and the *de facto* standard for folk and traditional music.*

In fact, this notation is specifically designed to meet the needs of traditional musicians; hundred of thousands (really!) of tunes in ABC formats are available. ABC is a simple text notation, originally designed for single-voice music but currently capable of producing complex polyphonic scores.

This is our usual music sample in ABC notation:

```
% music sample in ABC notation
X: 1
T: Music sample in ABC
M: C
L: 1/4
K: C
%
C D/E/ F/G/A/B/|c B/A/ G/F/E/D/|C/G/E/G/ C/G/E/G/|CECz|]
```

We typeset the score with these commands:

```
$ abcm2ps -c -O= sample-abc.abc
abcm2ps-8.14.1 (2018-11-15)
```



```
File sample-abc.abc
Output written on sample-abc.ps (1 page, 1 title, 20503 bytes)
$ ps2pdf sample-abc.ps
$ _
```

5.5 Program: [abc2svg](#)

[abc2svg](#) is a free and multiplatform scorewriter, released under the GNU GPL. It's basically [abcm2ps](#) rewritten in JavaScript.

Although [abc2svg](#) can be integrated in a [web-based editor](#), it's a command-line program. It reads an ABC source file and turns it to **xhtml**:

```
$ abc2svg file.abc > file.xhtml
$ _
```

The resulting **.xhtml** can then be loaded into any web browser and printed to PDF. [Google Chrome](#) or [Chromium](#) are the recommended browsers.

We can run the whole procedure non-interactively, entirely in the command line. This only works in GNU/Linux and macOS:

```
$ abc2svg tunes.abc > tunes.xhtml
$ chromium-browser --headless --print-to-pdf=tunes.pdf tunes.xhtml
... many log messages ...
... Written to file tunes.pdf.
$ _
```

However, in this case Chromium will add headers and footers to every page. To remove them and obtain a clean PDF file, run the command:

```
$ pdfcrop --margins "0 -9 0 -9" --clip tunes.pdf
PDFCROP 1.38, 2012/11/02 - Copyright (c) 2002-2012 by Heiko Oberdiek.
==> 79 pages written on 'tunes-crop.pdf'.
$ _
```

The same result can be obtained using [pdfpages](#) directly:

```
\documentclass{article}
\usepackage{pdfpages}
```



```

\def\tunes{tunes.pdf} % PDF file to trim

\begin{document}

\includepdf[pages=-,pagecommand={},%
width=\paperwidth,trim={0 0.9cm 0 0.9cm},clip]{\tunes}

\end{document}

```

5.6 Program: *PMW*

Philip's Music Writer (PMW) is a free and multiplatform scorewriter, released under the GNU GPL. From the PMW home page:

Philip's Music Writer (PMW) is a computer program for high quality music typesetting.

PMW uses a simple text notation for music input and produces output in PostScript.

This is our usual music sample in PMW format:

```

@ music sample in PMW

Heading "|Music sample in PMW"
Key C
Time 4/4

[stave 1 treble 1]
c d- e-; f-g-a-b-; | c' b- a-; g-f-e-d-; |
c-g-e-g-; c-g-e-g-; | c e c r |
[endstave]

```

We typeset the score with these commands:

```

$ pmw -includefonts sample-pmw.pmw
$ ps2pdf sample-pmw.ps
$ _

```


5.7 Program: *MUP*

MUP is a free and multiplatform scorewriter, released under the GNU GPL. From the MUP home page:

Mup takes a text file as input and produces very high quality PostScript output for printed music. It can handle both regular notation and tablature notation. It can also produce MIDI output.

This is our usual music sample in MUP format:

```
// music sample in MUP notation

header
  title "Music sample in MUP"

score
  time=4/4

music
  1: 4c; 8d bm; e ebm; f bm; g; a; b ebm;
  bar
  1: 4c+; 8b bm; a ebm; g bm; f; e; d ebm;
  bar
  1: 8c bm; g; e; g ebm; c bm; g; e; g ebm;
  bar
  1: 4c; e; c; r;
endbar
```

We typeset the score with these commands:

```
$ mup -F sample-mup.mup
Mup - Music Publisher Version 6.6
Copyright (c) 1995-2017 by Arkkra Enterprises.
Mup is free software. Use -l option to see license terms.
$ ps2pdf sample-mup.ps
$ _
```


5.8 Program: *MuseScore*

MuseScore is a free and multiplatform scorewriter, released under the GNU GPL. Unlike the previous programs, it's a desktop application; however, it can be conveniently used from the command line.

MuseScore uses its own file formats (*.mscz*, *.mscx*) but it can also import several other file formats; *MusicXML* is probably the most important.

We convert any supported file to PDF with this command:

```
$ musescore file.xml -o file.pdf
initScoreFonts 0x30d33c0
convert <file.xml> to <file.pdf>
setFirstInstrument: no instrument found for part 'P1'
$ _
```

5.9 Package *abc*

This package enables the inclusion of music in ABC notation in *L^AT_EX* sources; it's similar to *lilypond-book* or *lyluatex*, but targeting ABC.

ABC provides the *abc* environment and the *\abcinput* commands. The first embeds ABC music in the source, while the second includes an external ABC file:

```
\documentclass{article}
\usepackage[generate,ps2eps]{abc}
\thispagestyle{empty}

\begin{document}

This is an Irish reel:

\begin{abc}[name=julia,program={abcm2ps -O=}]
X:63
T: Julia Delaney's
M: C|
L: 1/8
R: reel
K: Ddor
|: dcAG ~F2EF|~E2 DE FD D2|dcAG FGAA|Addc d2 fe :|
   f2fe fagf |ecgc acgc   |f2fe fagf|edcG Add2  :|
\end{abc}

This is an Irish polka, slightly smaller:
```



```
\abcinput[program={abcm2ps -O=},width=0.9\abcwidth]
{breeches}

End of document.

\end{document}
```

The **generate** option (default) indicates that the ABC music should be generated by the external ABC typesetter. If the ABC music is not modified and it has already been generated, we could specify the **nogenerate** option. This is what we want to do when only the text is changed.

We typeset the source with this command:

```
$ pdflatex --shell-escape sample-abc.tex
...
Output written on sample-abc.pdf (1 page, 30937 bytes).
Transcript written on sample-abc.log.
$ _
```

This is an Irish reel:

Julia Delaney's



This is an Irish polka, slightly smaller:

Breeches Full of Stitches



End of document.



6 Guitar Tablatures

Let's go back to guitar stuff. Tablature, as already explained, is a form of musical notation for stringed instruments; making tablatures is basically the same process as making sheet music.

6.1 Program: **LilyPond**

In addition to sheet music, LilyPond can also easily typeset guitar tablatures. A minimal example (`lilytab.ly`):

```
\version "2.18.2"
\header { tagline = "" } % no footer
\paper { left-margin = 0\cm }

music = {
  \time 3/4
  c4 d e f g a b a b c'2 r4
}

\score {
  <<
    \new Staff { \clef "G_8" \music } % sheet music
    \new TabStaff { \music }          % tablature
  >>
}
```

We typeset the score with this command:

```
$ lilypond lilytab.ly
...
Converting to './lilytab.pdf'...
Success: compilation successfully completed
$ _
```



As you can see, the very same music line can be typeset as sheet music and as guitar tablature. Fretboard positions are automatically generated by LilyPond.

6.2 Program: [abc2xml.py](#)

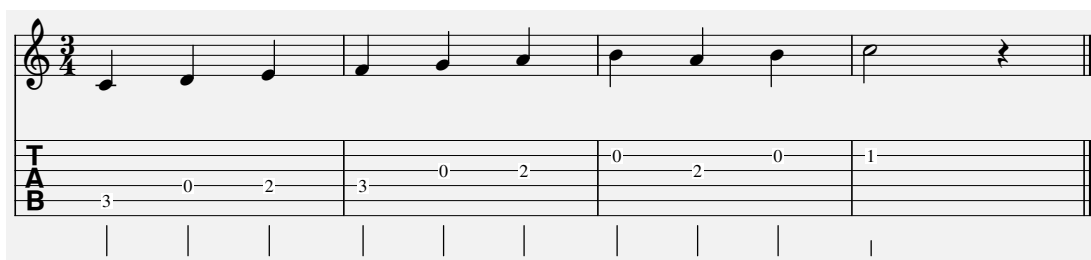
This Python program converts an ABC file to a [MusicXML](#) file containing tablature information; this file can then be typeset with any MusicXML-enabled application, like MuseScore.

A minimal tablature example ([abctab.abc](#)):

```
X: 1
M: 3/4
L: 1/4
K: C
%
V:1
CDE | FGA | BBB | c2z |]
V:2 clef=tab octave=-1
CDE | FGA | BBB | c2z |]
```

We typeset the score with these commands (error messages can be safely ignored):

```
$ abc2xml.py -f abctab.abc > abctab.xml
-- decoded from utf-8
-- skipped header: (field X,1)
-- done in 0.02 secs
$ musescore abctab.xml -o abctab.pdf
Jack appears to be installed on this system, so we'll use it.
initScoreFonts 0x25ba100
libpng warning: iCCP: known incorrect sRGB profile
convert <abctab.xml> to <abctab.pdf>
Error at line 18 col 16: no instrument found for part 'P1'
Error at line 138 col 16: no instrument found for part 'P2'
$ _
```

As you can see, the very same music line can be typeset as sheet music and as guitar tablature. Fretboard positions are automatically generated by `abc2xml.py`.

We can also use `lilypond` to typeset the ABC tablature. The ancillary program `musicxml2ly` converts MusicXML files to LilyPond format:

```
$ abc2xml.py -f abctab.abc > abctab.xml
-- decoded from utf-8
-- skipped header: (field X,1)
-- done in 0.02 secs
$ musicxml2ly abctab.xml
$ musicxml2ly: Reading MusicXML from abctab.xml ...
musicxml2ly: Converting to LilyPond expressions...
musicxml2ly: Converting to LilyPond expressions...
musicxml2ly: Output to 'abctab.ly'
$ lilypond abctab.ly
GNU LilyPond 2.18.2
Processing 'abctab.ly'
...
Converting to './abctab.pdf'...
Success: compilation successfully completed
$ _
```

Regrettably, `musicxml2ly` is not as robust as MuseScore's MusicXML import filter, and it may fail on complex music.

6.3 Package **guitartabs**

This package may be missing in less recent releases of TeX Live, so you might have to install it manually. It provides a **guitartabs** document class and a very simple syntax to denote strings, fret positions, and note length:


```

\documentclass{guitartabs}
\thispagestyle{empty}

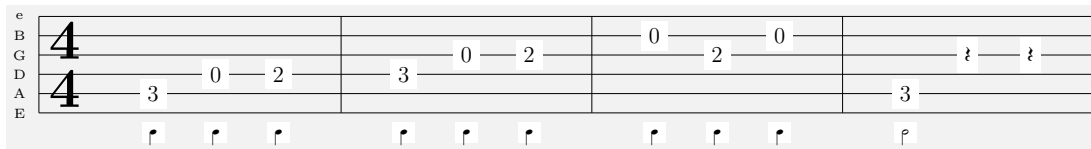
\begin{document}

\Large

\begin{tabline}{4}{4}{4}{E,A,D,G,B,e}
% bar 1
% note 1 of 3, string 5, fret 4, note length 1/4
\notel{1}{3}{5}{3}{4}
% note 2 of 3, string 4, fret 0, note length 1/4
\notel{2}{3}{4}{0}{4}
\notel{3}{3}{4}{2}{4}
% bar 2
\nextbar
\notel{1}{3}{4}{3}{4}
\notel{2}{3}{3}{0}{4}
\notel{3}{3}{3}{2}{4}
% bar 3
\nextbar
\notel{1}{3}{2}{0}{4}
\notel{2}{3}{3}{2}{4}
\notel{3}{3}{2}{0}{4}
% bar 4
\nextbar
\notel{1}{3}{5}{3}{2}
% rests
\restquarter{2}{3}
\restquarter{3}{3}
\end{tabline}

\end{document}

```



If you don't need the note length indications at the bottom of the tablature, use `\note` instead of `\notel`.

6.4 Using Guitar Pro Tablatures

A *de facto* standard for guitar tablatures is the [Guitar Pro](#) format, or more precisely a set file formats: **.gpX** (where X is 3, 4, 5, 6), **.gpx**, **.gtp**. Many sites offer free tablatures in one of these formats.

To print them, we can use [TuxGuitar](#) or MuseScore. TuxGuitar is a free and multiplatform tablature editor; it's a desktop application that can export tablatures in PDF format. Regrettably, it cannot work as a command-line program.

To import GuitarPro tablatures into MuseScore and export them as PDF, we can use the **-P** command line switch. It exports all parts to the PDF file; then it's up to the user to find out the pages that contain the tablature.



7 Putting It All Together

We have solved the first problem; now we know how to make music files in several different ways. Now it's time to solve the second problem: how to combine music files and \LaTeX .

7.1 Package [abc](#), Revisited

We met this package in Section 5.9, where it was used to include ABC music in a \LaTeX source. This package, however, is not limited to ABC. User-defined environments can provide support for virtually any external scorewriter; LilyPond support, however, is slightly bugged.

The following source defines the **mtx** and **pmw** environments. For each, the definition specifies the external program to run, its command line switches, and the file extension:

```
\documentclass{article}
\thispagestyle{empty}
\usepackage[generate,ps2eps]{abc}

% --- M-Tx support
\newenvironment{mtx}[1]{}
{\renewcommand{\normalabcoutputfile}{out-mtx}%
\abc[program=musixtex,options={-g},extension=mtx,#1]}
{\endabc}
\newcommand{\mtxinput}[2]{}%
```



```

\abcinput[program=musixtex,options={-g},extension=mtx,#1]{#2}}

% --- PMW support
\newenvironment{pmw}[1][
{\renewcommand{\normalabcoutputfile}{out-pmw}%
\abc[program=pmw,options={-includefont},extension=pmw,#1]}
{\endabc}
\newcommand{\pmwinput}[2][]{%
\abcinput[program=pmw,options={-includefont},extension=pmw,#1]{#2}}

\begin{document}

This document includes music excerpts written in different formats. It
uses \texttt{abc.sty} and defines new environments.

This is a short piece, typeset by M-Tx:

\mtxinput{sample-mtx}

The same piece, typeset by PMW:

\pmwinput{sample-pmw}

End of document.

\end{document}

```

This document includes music excerpts written in different formats. It uses `abc.sty` and defines new environments.

This is a short piece, typeset by M-Tx:

The same piece, typeset by PMW:

End of document.

A complete template that implements all environments is presented in Section [B.1](#).

7.2 Using *make*

Another way to make a \LaTeX document that includes music in different formats is by using a developer's tool called **make**. It's a program that takes care of what needs what, what needs to be converted first, what should be done if you modify something, and so on. **make** is normally used to compile programs.

Let's make a practical example. We have a \LaTeX document, `main.tex`, which includes three PDF files, `music1.pdf`, `music2.pdf`, and `music3.pdf`:

```
\documentclass{article}
\usepackage{graphicx}
\thispagestyle{empty}

\begin{document}

This document includes three music excerpts:

\includegraphics{music1}

\includegraphics{music2}

\includegraphics{music3}

End of document.

\end{document}
```

Let's suppose that the three PDF files are obtained from an M-Tx file, a LilyPond file, and an ABC file. We should convert the PDF files manually, then typeset `main.tex`. It's not a big deal, but what if you have dozens of music files, each of which must be converted with different commands? This task would soon grow tedious and difficult to manage.

Here **make** comes to the rescue. It uses a text file, called **Makefile**, which contains rules for building the document and the PDF files that it includes. This is a simple **Makefile** that can be used to compose (make!) `main.pdf`:

```
# Makefile for main.tex

FIGURES = music1.pdf music2.pdf music3.pdf

# The final document depends on main.tex and the figures
main.pdf: main.tex $(FIGURES)
    pdflatex main.tex

# music1.pdf depends on music1.mtx
# conversion commands follow
music1.pdf: music1.mtx
    musixtex music1.mtx ; pdfcrop musix1.pdf ; \
```



```

        /bin/mv musix1-crop.pdf music1.pdf

# music2.pdf depends on music2.ly
music2.pdf: music2.ly
    lilypond music2.ly; pdfcrop musix2.pdf ; \
    /bin/mv musix2-crop.pdf music2.pdf

# music3.pdf depends on music3.abc
music3.pdf: music3.abc
    abcm2ps -c -O= music3.abc; ps2pdf music3.ps; \
    pdfcrop musix3.pdf ; \
    /bin/mv musix3-crop.pdf music3.pdf

# end of Makefile

```

Let's see what it does. First of all, as you might have guessed the `#` character starts a comment; the rest of the line is ignored.

The line:

```
FIGURES = music1.pdf music2.pdf music3.pdf
```

creates a *variable*, that is a “name” (**FIGURES**) that “contains” the three file names **music1.pdf music2.pdf music3.pdf**. This variable will be referred to later on.

The lines:

```
main.pdf: main.tex $(FIGURES)
    pdflatex main.tex
```

declare that **main.pdf** is a *target* that depends on **main.tex** and on the three files denoted by the **FIGURES** variable. The second line specifies the command that must be run to make **main.pdf**. This line starts with a TAB character, not with spaces: this is important!

Then we have three sections, one for each music file. Each section tells **make** what to do to compose the PDF figure. For instance, the section:

```
music1.pdf: music1.mtx
    musixtex music1.mtx ; pdfcrop musix1.pdf ; \
    /bin/mv musix1-crop.pdf music1.pdf
```

states that **music1.pdf** is a target that depends on **music1.mtx**; the following lines specify the commands to create **music1.pdf**. Commands are separated by the `;` character, while `\` indicates that the command continues to the next line.

To make **main.pdf**, we simply run the command:


```
$ make
...
$ _
```

in the same directory where we saved **Makefile**. **make** will make the three figures first, then **main.pdf** that depends on them.

If we modify one of the files, **make** will take care of dependencies and rebuild the final target. For example, if we modify **music2.ly**, **make** will rebuild **music2.pdf** first, then **main.pdf** that depends on **music2.pdf**.

In my opinion, this approach is the most flexible. This guide was compiled using **make** and a pretty long **Makefile**.

♪ ♪ ♪ ♪ ♪ ♪ ♪ ♪

8 The End

That's it, dear fellow musicians: I really hope that this guide will be useful to you. Please help me improve this document: for any suggestions, comments, or contributions, please feel free to contact me by [email](#). I'd like to receive feedback, especially if you use this document in education.

This document is copyleft © Guido Gonzato, PhD, and released under the [GNU Free Documentation Licence](#).

♪ ♪ ♪ ♪ ♪ ♪ ♪ ♪

A List of Packages and Programs

This manual uses many L^AT_EX packages and includes PDF output produced by several programs, all of which are free and open source. Commercial programs were intentionally left out.

The page <https://www.ctan.org/topic/music> is an important starting point for those interested in combining L^AT_EX and music. It's a list of L^AT_EX packages for typesetting music and related stuff. The following is the complete list of packages and programs I used to make this manual, in order of appearance.

- CTAN Music page. “This topic contains packages for typesetting music and related stuff.”
<https://www.ctan.org/topic/music>.
- Package: musicography.
<https://ctan.org/pkg/musicography>
- Package: leadsheets.
<https://ctan.org/pkg/leadsheets>
- Package: lilyglyphs.
<https://ctan.org/pkg/lilyglyphs>
- Font: Bravura.
<https://www.smubl.org/fonts>
- Package: guitar.
<https://ctan.org/pkg/guitar>
- Package: gtrcrd.
<https://ctan.org/pkg/gtrcrd>
- Package: songs.
<https://ctan.org/pkg/songs>
- Package: musixguit.
<https://ctan.org/pkg/musixguit>
- Package: songbook.
<https://ctan.org/pkg/songbook>
- Program: Chordii.
<https://www.vromans.org/projects/Chordii>

- Package: gchords.
<https://ctan.org/pkg/gchords>
- Package: guitarchordschemes.
<https://ctan.org/pkg/guitarchordschemes>
- Package: guitartabs.
<https://ctan.org/pkg/guitartabs>
- Page: MusiXTeX and Related Software.
<https://icking-music-archive.org/software/htdocs/htdocs.html>
- Package: MusiXTeX.
<https://ctan.org/pkg/musixtex>
- Package: M-Tx.
<https://ctan.org/pkg/m-tx>
- Package: Gregoriotex.
<https://ctan.org/pkg/gregoriotex>
<http://gregorio-project.github.io/gregoriotex>
<http://gregorio-project.github.io>
- Program: LilyPond.
<http://lilypond.org>
- Package: Lyluatex.
<https://ctan.org/pkg/lyluatex>
- Programs: **abcm2ps**, **and2svg**.
<http://moinejf.free.fr>
- Program: PMW, Philip's Music Writer.
<http://people.ds.cam.ac.uk/ph10/pmw.html>
- Program: MUP.
<http://www.arkkra.com>
- Program: MuseScore.
<http://musescore.org>
- Package: Abc.
<https://ctan.org/pkg/abc>
- Program: abc2xml.ly.
<https://wim.vree.org/svgParse/abc2xml.html>

B Examples

B.1 A Complete `abc` Example

This source defines environments for M-Tx, PMW, LilyPond, and MUP; it also redefines the `abc` environment as `ABC`. This is necessary for technical reasons. Lilypond sources must begin with a double line `\version "2.18.2"`; this is required to avoid a bug.

```
% typeset with:
% pdflatex -shell-escape sample-abc-all.tex

\documentclass{article}
\thispagestyle{empty}
\usepackage[generate,ps2eps]{abc}

% --- M-Tx support
\newenvironment{mtx}[1][ ]
{\renewcommand{\normalabcoutputfile}{out-mtx}%
\abc[program=musixtex,options={-g},extension=mtx,#1]}
{\endabc}
\newcommand{\mtxinput}[2][ ]{%
\abcinput[program=musixtex,options={-g},extension=mtx,#1]{#2}}

% --- PMW support
\newenvironment{pmw}[1][ ]
{\renewcommand{\normalabcoutputfile}{out-pmw}%
\abc[program=pmw,options={-includefont},extension=pmw,#1]}
{\endabc}
\newcommand{\pmwinput}[2][ ]{%
\abcinput[program=pmw,options={-includefont},extension=pmw,#1]{#2}}

% --- LilyPond support
% !!! BUG: the LilyPond source must begin with a \null command
\newenvironment{lily}[1][ ]
{\renewcommand{\normalabcoutputfile}{out-lily}%
\abc[program=lilypond,options={-d backend=eps},extension=ly,#1]}
{\endabc}
\newcommand{\lilyinput}[2][ ]{%
\abcinput[program=lilypond,options={--ps},extension=ly,#1]{#2}}

% --- MUP support
\newenvironment{mup}[1][ ]
{\renewcommand{\normalabcoutputfile}{out-mup}%
\abc[program=mup,options={-F},extension=mup,#1]}
{\endabc}
```



```

\newcommand{\mupinput}[2][ ]{%
\abcinput[program=mup,options={-F},extension=mup,#1]{#2}}

% --- ABC must be redefined
\newenvironment{ABC}[1][ ]
{\renewcommand{\normalabcoutputfile}{out-ABC}%
\abc[program=abcm2ps,options={-O=},extension=abc,#1]}
{\endabc}
\newcommand{\ABCinput}[2][ ]{%
\abcinput[program=abcm2ps,options={-O=},extension=abc,#1]{#2}}

\begin{document}

This document includes music excerpts written in several formats. It
uses \texttt{abc.sty} and defines new environments.

This is a short piece, typeset by M-Tx:

\begin{mtx}
Title: Music sample in M-Tx
Style: Solo
Meter: C
Width: 160mm

c4 d8 e f g a b | c4 b8 a g f e d | c8 g+ e g c- g+ e g | c4- e c r |
\end{mtx}

The same piece, typeset by LilyPond:

\begin{lily}
% twice - it's required to avoid a bug
\version "2.18.2"
\version "2.18.2"

\header {
  title = "Music sample in LilyPond"
  tagline = "" % no footer
}

\relative c' {
  \time 4/4
  \clef treble
  c4 d8 e f8 g a b | c4 b8 a g8 f e d |
  c8 g' e g c,8 g' e g | c,4 e c r \bar "|."
}
\end{lily}

```


The same piece, typeset by PMW:

```
\begin{pmw}
Heading "|Music sample in PMW"
Key C
Time 4/4

[stave 1 treble 1]
c d- e-; f-g-a-b-; | c' b- a-; g-f-e-d-; |
c-g-e-g-; c-g-e-g-; |c e c r |
[endstave]
\end{pmw}
```

The same piece, typeset by MUP:

```
\begin{mup}
// music sample in MUP notation

header
  title "Music sample in MUP"

score
  time=4/4

music
  1: 4c; 8d bm; e ebm; f bm; g; a; b ebm;
  bar
  1: 4c+; 8b bm; a ebm; g bm; f; e; d ebm;
  bar
  1: 8c bm; g; e; g ebm; c bm; g; e; g ebm;
  bar
  1: 4c; e; c; r;
  endbar
\end{mup}
```

The same piece, typeset by abcm2ps:

```
\begin{ABC}
X: 1
T: Music sample in ABC
M: 4/4
L: 1/4
K: C
%
C D/E/ F/G/A/B/|c B/A/ G/F/E/D/|C/G/E/G/ C/G/E/G/|CECz|]
\end{ABC}
```

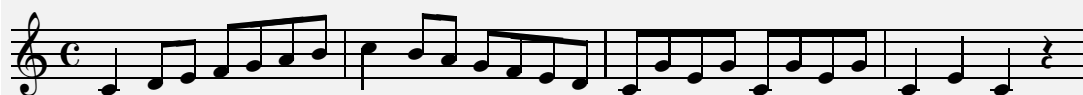


```
\end{document}
```

This document includes music excerpts written in several formats. It uses `abc.sty` and defines new environments.

This is a short piece, typeset by M-Tx:

Music sample in M-Tx



The same piece, typeset by LilyPond:

Music sample in LilyPond



The same piece, typeset by PMW:

Music sample in PMW



The same piece, typeset by MUP:

Music sample in MUP



The same piece, typeset by `abcm2ps`:

Music sample in ABC



B.2 A Complete Songbook Example

The following source is a minimal template for songbooks. It uses the `gchords` and `guitar` packages (Sections 4.1 and 3.1). The resulting PDF has its own page numbers (don't get confused!) and is included in this document with `\includepdf`.

```
\documentclass[11pt]{article}
\usepackage{graphicx}
\usepackage{gchords}
\usepackage{guitar}

\begin{document}

\title{A Minimal Songbook}
\author{Guido Gonzato}

\maketitle

\tableofcontents

\bigskip
\hrule

% -----

\section{For He's a Jolly Good Fellow}

According to the 1998 Guinness World Records, this is the second most
recognized song in the English language.

\def\numfrets{4}
\chords{
\chord{t}{n,p3,p2,n,p1,n}{C}
\chord{t}{p3,p2,n,n,n,p3}{G}
\chord{t1}{n,p2,p2,p1,n,n}{F}
}

\medskip

\includegraphics[width=\textwidth]{fellow}

% -----

\section{Happy Birthday To You}
```


According to the 1998 Guinness World Records, it is the most recognized song in the English language, followed by ‘‘For He’s a Jolly Good Fellow’’.

```
\def\numfrets{4}
\chords{
\chord{t1}{n,p2,p2,p1,n,n}{F}
\chord{t}{n,p3,p2,n,p1,n}{C}
\chord{t}{n,p3,p2,p3,p1,n}{C7}
\chord{t1}{n,n,p2,p2,p2,n}{B$\flat$}
}

\medskip

\includegraphics[width=\textwidth]{happyb}

% -----

\section{Twinkle, Twinkle Little Star}

This is a popular English lullaby.

\def\numfrets{4}
\chords{
\chord{t}{n,p3,p2,n,p1,n}{C}
\chord{t1}{n,p2,p2,p1,n,n}{F}
\chord{t}{x,n,n,p2,p3,p1}{Dm}
\chord{t}{p3,p2,n,n,n,p3}{G}
}

\includegraphics[width=\textwidth]{twinkle}

\medskip

\begin{guitar}

[C]Twinkle, twinkle, [F]little [C]star,
[Dm]How I [C]wonder [G]what you [C]are!
[C]Up a [Dm]bove the [C]world so [G]high,
[C]Like a [Dm]diamond [C]in the [G]sky.
Twinkle{\ldots}

[C]When this blazing [F]sun is [C]gone,
[Dm]When he [C]nothing [G]shines u[C]pon,
[C]Then you [Dm]show your [C]little [G]light,
[C]Twinkle, [Dm]twinkle, [C]through the night.
Twinkle{\ldots}
```



```
[C]Then the [F]traveller in the [C]dark
[Dm]Thanks you [C]for your [G]tiny [C]spark;
[C]He could [Dm]not see [C]where to [G]go,
[C]If you [Dm]did not [C]twinkle [G]so.
Twinkle{\ldots}

[C]In the dark blue [F]sky you [C]keep,
And [Dm]often [C]through my [G]curtains [C]peep,
[C]For you [Dm]never [C]shut your [G]eye
[C]Till the [Dm]sun is [C]in the [G]sky.
Twinkle{\ldots}

[C]As your bright and [F]tiny [C]spark
[Dm]Lights the [C]traveller [G]in the [C]dark,
[C]Though I [Dm]know not [C]what you [G]are,
[C]Twinkle, [Dm]twinkle, [C]little [G]star.
Twinkle{\ldots}

\end{guitar}

\end{document}
```


A Minimal Songbook

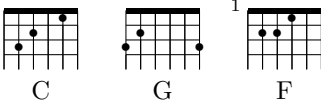
Guido Gonzato
January 6, 2019

Contents

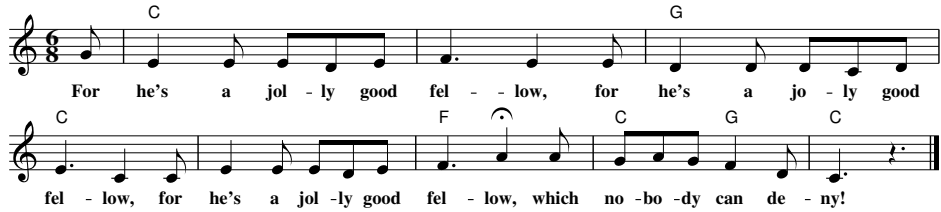
1	For He’s a Jolly Good Fellow	1
2	Happy Birthday To You	1
3	Twinkle, Twinkle Little Star	2

1 For He’s a Jolly Good Fellow

According to the 1998 Guinness World Records, this is the second most recognized song in the English language.



C G F



2 Happy Birthday To You

According to the 1998 Guinness World Records, it is the most recognized song in the English language, followed by “For He’s a Jolly Good Fellow”.

1

F C C7 B \flat

Hap - py birth - day to you, hap - py birth - day to you, hap - py
 birth - day dear Gui - do, hap - py birth - day to you!

3 Twinkle, Twinkle Little Star

This is a popular English lullaby.

1

C F Dm G

Twin - kle, twin - kle lit - le star, how I won - der what you are!
 Up a - bove the world so high, like a dia - mond in the sky.
 Twin - kle, twin - kle lit - le star, how I won - der what you are!

C F C
 Twinkle, twinkle, little star,
 Dm C G C
 How I wonder what you are!
 C Dm C G
 Up above the world so high,
 C Dm C G
 Like a diamond in the sky.
 Twinkle...
 C F C
 When this blazing sun is gone,
 Dm C G C
 When he nothing shines upon,

C Dm C G
Then you show your little light,
C Dm C
Twinkle, twinkle, through the night.
Twinkle...

C F C
Then the traveller in the dark
Dm C G C
Thanks you for your tiny spark;
C Dm C G
He could not see where to go,
C Dm C G
If you did not twinkle so.
Twinkle...

C F C
In the dark blue sky you keep,
Dm C G C
And often through my curtains peep,
C Dm C G
For you never shut your eye
C Dm C G
Till the sun is in the sky.
Twinkle...

C F C
As your bright and tiny spark
Dm C G C
Lights the traveller in the dark,
C Dm C G
Though I know not what you are,
C Dm C G
Twinkle, twinkle, little star.
Twinkle...

B.3 A sample Makefile

This is the **Makefile** that was used to typeset the previous example. The **clean** target is used to clean up all temporary files.

```
# Makefile for songbook.tex

FIGURES = fellow.pdf happyb.pdf twinkle.pdf

songbook: songbook.tex $(FIGURES)
    pdflatex songbook.tex

fellow.pdf: fellow.abc
    abcm2ps -c -O= fellow.abc; \
    ps2pdf fellow.ps; \
    pdfcrop fellow.pdf; \
    /bin/mv -f fellow-crop.pdf fellow.pdf

happyb.pdf: happyb.abc
    abcm2ps -O= happyb.abc; \
    ps2pdf happyb.ps; \
    pdfcrop happyb.pdf; \
    /bin/mv -f happyb-crop.pdf happyb.pdf

twinkle.pdf: twinkle.abc
    abcm2ps -O= twinkle.abc; \
    ps2pdf twinkle.ps; \
    pdfcrop twinkle.pdf; \
    /bin/mv -f twinkle-crop.pdf twinkle.pdf

clean:
    /bin/rm -f .*~ *~ *aux *bak *lo? *to? *out *tmp *bbl *ps
```