

Chordii

User Guide

Johan Vromans

Version 4.5.3, November 2015



Table of Contents

Welcome to Chordii!.....	4
An Example.....	5
How do I use Chordii?.....	7
Chord names.....	8
Directives.....	9
Song related directives.....	9
Output related directives.....	10
Preferences.....	11
Defining chords.....	12
Command line options.....	13
About Chordii.....	15
Release notes.....	16
4.5.2, October 2015.....	16
4.5.1, June 2013.....	16
4.5, June 2013.....	16
4.4, September 2012.....	16
4.3, December 2009.....	16
4.2, June 2008.....	16
4.1, March 2008.....	16
4.0, November 2007.....	16
Appendix A. How do I install Chordii on my machine?.....	18
From your system repository.....	18
From the SourceForge web site.....	18
Building from source.....	18
Appendix B. Where to find Chordii files.....	19
Appendix C. Some Future plans.....	20
PDF generation.....	20

Copyright 2007,2015 Johan Vromans. All rights reserved.

This document is based on original work of Martin Leclerc and Mario Dorion.



This work is licensed under the Creative Commons Attribution-Noncommercial-Share Alike 3.0 Unported License.

To view a copy of this license, visit
<http://creativecommons.org/licenses/by-nc-sa/3.0/>

UNICODE.....20

Welcome to Chordii!

Chordii, pronounced *chord-ee-ee*, is a utility that was first created by lazy guitarists who got tired of turning pages in the middle of the songs they liked.

Chordii takes one or more input files containing the lyrics and chords of one or more songs and produces a print (PostScript) version of these songs.

The output has the following characteristics:

- titles and sub-titles have been centered,
- the lyrics appear in the font and size of your choice,
- all chords names appear right above the right lyrics,
- all chords used in a song appear as grids at the bottom of the page.

Optionally, you can also:

- generate an index of your songs,
- have the pages numbered,
- have chords transposed up or down,
- print in 2-up or 4-up modes (multiple pages per printed sheet) and,
- insert tablature and comments.

You have a great many options on the final appearance of your songs. All of them are described in section 'Directives', page 9. For now, please read on to better understand what Chordii can do for you.

An Example

This is an example of a 'source' file for Chordii, i.e. a file that is meant to be processed by Chordii. You can use any convenient text editor to create it.

```
{title:Swing Low Sweet Chariot}
{st:Traditional}

{start_of_chorus}
Swing [D]low, sweet [G]chari[D]ot,
Comin' for to carry me [A7]home.
Swing [D7]low, sweet [G]chari[D]ot,
Comin' for to [A7]carry me [D]home.
{end_of_chorus}
```

```
I looked over Jordan, and what did I see,
    Comin' for to carry me home.
A band of angels comin' after me,
    Comin' for to carry me home.
```

```
{c:Chorus}
```

```
If you get there before I do,
    Comin' for to carry me home.
Just tell my friends that I'm a comin' too.
    Comin' for to carry me home.
```

```
{c:Chorus}
```

```
I'm sometimes up and sometimes down,
    Comin' for to carry me home.
But still my soul feels heavenly bound.
    Comin' for to carry me home.
```

```
{c:Chorus}
```

As mentioned, this file can be created using any text editor you like. All characters present in the ISO 8859-1 (Latin-1) character set can be used. We will explain the content of that file later. But first, let's look at the result, on the next page.

Swing Low Sweet Chariot
Traditional

D
G D
 Swing low, sweet chariot,
A7
 Comin' for to carry me home.
D7
G D
 Swing low, sweet chariot,
A7
D
 Comin' for to carry me home.

I looked over Jordan, and what did I see,
 Comin' for to carry me home.
 A band of angels comin' after me,
 Comin' for to carry me home.

Chorus


If you get there before I do,
 Comin' for to carry me home.
 Just tell my friends that I'm a comin' too.
 Comin' for to carry me home.

Chorus


I'm sometimes up and sometimes down,
 Comin' for to carry me home.
 But still my soul feels heavenly bound.
 Comin' for to carry me home.

Chorus


A7




D



D7



G



Example output of 'Swing Low, Sweet Chariot'.

How do I use Chordii?

Using Chordii is simple.

Four steps are required:

1. Type in the lyrics of the songs: this can be done with any convenient text editor.

```
Swing low, sweet chariot,
```

2. Insert the chord names between square brackets throughout the lyrics. The chord name will appear right above the letter that follows the closing bracket

```
Swing [D]low, sweet [G]chari[D]ot,
```

3. Insert any directive you want to give to Chordii: titles, sub-titles, chorus markers, comments, etc. Directives are described in section ‘Directives’, page 9.

```
{title:Swing Low Sweet Chariot}  
{st:Traditional}
```

When you are finished, save the file as ‘swinglow.cho’ to your disk.

4. Invoke Chordii with your file as an argument. This will produce the PostScript output. You can save this in a file for future printing. Invoking Chordii is described in more detail in section ‘Command line options’, page 13.

```
$ chordii -o swinglow.ps swinglow.cho
```

If you have a PostScript capable printer, you can print it directly:

```
$ lp swinglow.ps
```

You can also preview the result using a PostScript previewer, e.g. Ghostscript or GView. It is also possible to print it from the viewer.

When your printer does not support PostScript directly, you can convert the output to PDF using a suitable tool, e.g. ps2pdf.

```
$ ps2pdf swinglow.ps swinglow.pdf
```

You can then use Acrobat Reader to view the document, and to print it.

Lines in the input file that start with a ‘#’ are completely ignored. You can use this to embed notes and remarks to yourself in the file. For example,

```
# Hmm. Not sure about the G. Maybe a G7 sounds nicer?  
Swing [D]low, sweet [G]chari[D]ot,
```

Chord names

As you can see from the example, chord names are written between [] and placed in the lyrics, immediately preceding the text they should be placed on top of. For example,

Swing [D]low, sweet [G]chari[D]ot,

will be printed as

D
G D
 Swing low, sweet chari ot,

Actually you can put anything between the [] and Chordii will happily put that on top of the lyrics, but you will get a warning that that chord is not known to Chordii.

If you want to make use of Chordii's transpose feature it is necessary that the chord can be understood. For this, it has to obey some rules.

First, it must start with a note letter, i.e., one of 'A', 'B', 'C', 'D', 'E', 'F', and 'G'. This must be an upper-case letter.

The note letter may be followed by a flat symbol, for which we use the letter 'b', or a sharp symbol, '#'.

Then comes the chord modifier, if any. For example, 'm' for a minor chord, '7' for a 7th chord, and so on. In fact, this modifier may be arbitrary characters except for a slash, '/'.

To construct power chords, you can join two or more of these chords with a slash, '/'.

Some examples of valid chord names:

D Dm D#maj Eb+ Eb(add9) Em/B F+7+11

Chordii knows about 360 chords already, and you can add your own if you like.

Directives

A directive is a string of text that appears between curly brackets, { }, and has a special meaning to Chordii. It contains a message that will affect the way Chordii processes your file. Examples of directive uses would be: changing the fonts, marking a chorus, and defining a title.

Directives must be alone on a line. Blanks before the opening bracket and after the closing bracket are not significant. Blanks inside a directive are ignored. Many directives have shorter alternatives for convenience.

Directives fall into two categories: those that specify characteristics of the song being processed, like the song title, and those that control how the output should look like.

Song related directives

Directive	Description
<code>{new_song}</code> <code>{ns}</code>	Marks the beginning of a new song. It enables you to put multiple songs in one file. It is not required for the first (or only) song.
<code>{title: text}</code> <code>{t: text}</code>	Specifies the title of the song. It will appear centered at the top of the first page, and at the bottom of every other page, accompanied there by the page number, within the current song.
<code>{subtitle: text}</code> <code>{st: text}</code>	Specifies a text to be printed right below the title. You can specify more than one subtitle.
<code>{start_of_chorus}</code> <code>{soc}</code> <code>{end_of_chorus}</code> <code>{eoc}</code>	Indicates a chorus. The complete chorus will be highlighted by a margin bar, to be easily located by the player.
<code>{comment: text}</code> <code>{c: text}</code>	Prints the text in a grey box. Useful to call a chorus, for example.
<code>{comment_italic: text}</code> <code>{ci: text}</code>	Prints the text in an italic font. Well not really, it will print the text in the font used for printing the chord names, which is normally italic unless you specified a different font using the <code>chord_font</code> directive.
<code>{comment_box: text}</code> <code>{cb: text}</code>	Will print the text inside a bounding box.
<code>{start_of_tab}</code> <code>{sot}</code> <code>{end_of_tab}</code> <code>{eot}</code>	Indicates a piece of text that will be printed 'as is', using a monospace font. This can be used to enter 'tab' information where character positioning is crucial.
<code>{define: name</code> <code>base-fret offset</code> <code>frets str1...str6}</code>	Defines a new chord. See section 'Defining chords', page 12.

Output related directives

Some output related directives can also be specified on the command line when invoking Chordii.

Directive	Command line	Description
{textfont: <i>fontname</i> }	-T <i>fontname</i>	Sets the font used to print texts to the specified name. The name must be known to your PostScript printer.
{textsize: <i>number</i> }	-t <i>number</i>	Sets the size, in points, of the font used to print texts.
{chordfont: <i>fontname</i> }	-C <i>fontname</i>	Sets the font used to print chords to the specified name. The name must be known to your PostScript printer.
{chordsize: <i>number</i> }	-c <i>number</i>	Sets the size, in points, of the font used to display chords.
{no_grid} {ng}		Suppresses printing of the chord grids for the current song.
{grid} {g}		Enables printing of the chord grids for the current song (subject to the limitation caused by the usage of the -g option). This directive will override the -G option for the current song only.
{titles: <i>flush</i> }		If <i>flush</i> is 'left', titles are flushed left to the page. If <i>flush</i> is 'center', titles are centered to the page. This is the default behaviour.
{new_page} {np}		When printing in 2-up or 4-up mode, forces a logical page break. Otherwise this is the same as a physical page break.
{new_physical_page} {npp}		Forces a physical page break.
{columns: <i>number</i> }		Specifies the number of columns on the pages for the current song.
{column_break} {colb}		Forces a column break. The next line of the song will appear in the next available column, at the same height as the last 'columns' statement if still on the same page, or at the top of the page otherwise.
{pagetype: <i>type</i> }	-P <i>type</i>	Species the page type, e.g. 'a4' or 'letter'.

For more information on invoking Chordii and using command line options, see section 'Command line options', page 13.

Preferences

When running the Chordii program it will first try to read directives from a personal preferences file. This file is named `.chordrc`¹ and resides in your home directory. This offers a convenient way to preset output preferences and store your personal chord definitions.

An example of a preferences file:

```
{pagetype: a4}
{chordfont: Helvetica-LightOblique}
{chordsize: 9}
{textfont: Garamond-Light}
{textsize: 11}
{no_grid}
{titles: left}
{define: D4 base-fret 0 frets - - 0 0 3 -}
```

You can set environment variable `CHORDIIRC` (or `CHORDRC`²) to override the standard location of the preferences file.

1 This is for compatibility with existing `.chordrc` files for the old Chord program.

2 So is this.

Defining chords

Chordii knows of about 360 chords. However, there are many more chords possible, and many standard chords may be used with different voicings. So we put in a facility for people to define their own chords. This facility will also let you redefine already defined chords.

To define your own chords, and chord voicings, use the `{define}` directive.

```
{define: name base-fret offset frets str ... str}
```

For example:

```
{define: Am/C base-fret 0 frets 0 3 2 2 1 0 }
```

The keyword 'base-fret' indicates that the number that follows is the *offset*, i.e., the first fret that is to be displayed when representing this chord. Default value is '1'. The keyword 'frets' introduces 6 values. These are the fret numbers for each of the strings where a finger must be placed. These values are relative to the given base-fret offset. Keywords 'base-fret' and 'frets' are mandatory.

A value of '-', 'X' or 'x' indicates a string that is not played. A value of 0 for a given string means it is to be played open, and will be marked by a small open circle above the string in the grid. The strings are enumerated from left (lowest string) to right (highest string). On output, a chord defined in the users preferences file will have a small asterisk near its grid, and chords defined within a song will have two small asterisks.

At the beginning of every song, the default chords are reloaded and the users preferences file is reread. Chord definitions inside the text of a song apply to that song only.

Command line options

Chordii is a command line program. This means that you need a command prompt to invoke it. A command prompt can be obtained by starting the Terminal program (on GNU/Linux and Mac OS/X) or the Command Prompt on Microsoft Windows.

If Chordii has been properly installed, it can be run by typing its name at the command prompt, followed by its arguments. For example,

```
$ chordii -t 14 -s 6 -o swinglow.ps swinglow.cho
```

In this example, '\$' is the command prompt. 'chordii' is the program name. The rest are arguments to the program. Of these, 'swinglow.cho' is the file to process and the other things are command line options. Chordii understands single character options, as shown above, and also long option names that are usually better to remember. For example, the above command using long option names would be:

```
$ chordii --text-size=14 --chord-grid-size=6 --output=swinglow.ps swinglow.cho
```

The table below shows the short and long options, and their meanings. Column 'Arg' indicates that the option requires a value. For more information on command line options, consult the documentation for GNU 'getopt'.

Long option name	Short	Arg	Description
	-		A lone dash can be used to instruct Chordii to process the contents of standard input.
--2-up	-2		2-up printing. Prints two logical pages per physical page.
--4-up	-4		4-up printing. Prints four logical pages per physical page.
--about	-A		Prints the 'About Chordii...' message.
--chord-font	-C	<input checked="" type="checkbox"/>	Sets the font used to print chords. The font must be known to your PostScript printer.
--chord-grid-size	-s	<input checked="" type="checkbox"/>	Sets the size of the chord grids.
--chord-grids-sorted	-S		Prints chords grids alphabetically.
--chord-size	-c	<input checked="" type="checkbox"/>	Sets the size, in points, of the font used to display chords.
--dump-chords	-D		Generates a PostScript Chord Chart of all internally known chords as well as chords defined in the preferences file. Chords defined in the preferences file are identified with a small asterisk after the chord grid.
--dump-chords-text	-d		Generates a textual representation of all internally known chords as well as chords defined in the preferences file. Chords defined in the preferences file are identified with the '(local)' caption. The result is suitable for use in a preferences file.

Long option name	Short	Arg	Description
--even-pages-number-left	-L		Places the odd and even page numbers in the lower right and left corners respectively (for two-sided output). The default is all page numbers on the right side.
--help	-h		Prints a short options summary.
--lyrics-only	-l		Prints only the lyrics of the song.
--nashville	-N		Enables basic support for Nashville chords.
--no-chord-grids	-G		Suppresses printing of the chord grids for all input files. This can be re-enabled for any particular song with the <code>{grid}</code> directive.
--no-easy-chord-grids	-g		Suppresses printing of grids for 'easy' chords. Any chord in its major, minor, 7th or minor 7th is considered 'easy', while everything else is considered 'tough'. All chords defined in the preferences or in the input file are considered 'tough' as well.
--output	-o	<input checked="" type="checkbox"/>	Sends output to the indicated file. An existing file will be overwritten without warning.
--page-number-logical	-n		
--page-size	-P	<input checked="" type="checkbox"/>	Specify the paper size. Recognized values are 'a4' for ISO standard A4 paper (210mm × 297mm), 'a5' for ISO standard A5 paper (143mm × 210mm), and 'letter' (or 'us') for US Letter paper (8.5in × 11in).
--single-space	-a		Automatically single spaces lines that have no chords.
--start-page-number	-p	<input checked="" type="checkbox"/>	Numbers the pages consecutively starting with <code>first_page</code> (e.g. 1). Without this option, each song restarts the page numbering at 1, and page numbers are only put on subsequent pages of multiple page songs.
--text-font	-T	<input checked="" type="checkbox"/>	Sets the font used to print text to the specified name. The font must be known to your PostScript printer.
--text-size	-t	<input checked="" type="checkbox"/>	Sets the size, in points, of the font used to display the lyrics to the specified integer value. The title line is displayed using that point size + 5. The sub-title is displayed using that point size - 2.
--toc	-i		Generates a table of contents with the song titles and page numbers. It implies page numbering through the document.
--transpose	-x	<input checked="" type="checkbox"/>	Sets up transposition to that number of semitones. A positive argument produces sharps, while a negative argument will produce flats.
--version	-V		Prints the program version.
--vertical-space	-w	<input checked="" type="checkbox"/>	Inserts extra vertical space, in points, between lines.

About Chordii

Chordii is originally written by Martin Leclerc and Mario Dorion, somewhere between 1992 and 1995. They have given up interest.

In 2007, Johan Vromans and Adam Monsen decided to revive it. After having tried, and failed, to contact the original authors, they set up a new project at SourceForge named Chordie. At that time Johan already made several modifications to the original program for his own use. These changes were merged into the existing program, the code base was cleaned and modernized, the documentation overhauled and revised, and in November 2007 Chordie version 4.0 was released.

Due to possible confusion with the Chordie.com web site, in March 2008 the name was changed upon request to Chordii.

Chordii version 4 and later is Copyright 2007 The Chordii Project

Chord was originally licensed under the GPL, but with some restrictions. This imposed restrictions to Chordii as well. By the end of 2009, Johan managed to track down the original authors and got permission to release a new version of Chord, 3.6.4, under an unrestricted GPL license. After that, Chordii was rebased on Chord 3.6.4 effectively removing the license restrictions.

So both Chord and Chordii are now GPL version 2 or later.

The Chordii Project: <http://www.sourceforge.net/projects/chordii/>

Chordii home page: <http://chordii.sourceforge.net/>

Source code repository: <https://chordii.svn.sourceforge.net/>

GNU General Public License: <http://www.gnu.org/copyleft/gpl.html>

Release notes

4.5.2, October 2015

Document transpose behaviour with regards to sharps and flats. For convenience, allow the value zero for the ‘--transpose’ command line argument.

It is now possible to write arbitrary text between '[' and ']' . This can be used to place directives over the lyrics, like ‘Forte’, ‘Pianissimo’, and ‘Rit.’. Chordii will still print a warning, though.

Fix crashes on Microsoft Windows when environment variable HOME is not set.

Better support for MS-Dos (CR/LF) input files.

Minor bug fixes.

4.5.1, June 2013

Minor bug fixes.

4.5, June 2013

Allow reading input from standard input.

Configure option ‘--with-latin2’ for ISO 8859.2 character set support.

Support for A5 paper size.

Option ‘--chord-grids-sorted’ to print the chords in alphabetical order.

Minor bug fixes.

4.4, September 2012

This version was never officially released.

Long option names and ‘--vertical-space’ option.

Compile-time support for ISO 8859.2 character set.

Minor bug fixes.

4.3, December 2009

No new features, just an update following the rebase on Chord 3.6.4..

4.2, June 2008

This release introduces the first port of Chordii to Microsoft Windows.

4.1, March 2008

This release mainly introduces the new name, Chordii.

The settings file is now `.chordrc`, to be compatible with existing setting files from the old program.

Manual pages for `chordii` and `a2crd` are now included.

It is now possible to build a Debian package from the kit.

4.0, November 2007

This is the first official release of Chordie, as the revivification of the old Chord program.

Added a `-P` command line switch to select the paper format. Supported values are `a4` and `letter`.

Added a `pagetype` directive, to be placed in the `.chordrc`. Supported values are `a4` and `letter`.

Added a `titles` directive, to control title alignment. Supported values are `left` and `center`.

Modernized the build system, using the GNU built tools.

It is now possible to build an RPM package from the kit.

Appendix A. How do I install Chordii on my machine?

From your system repository

Many Linux distributions have Chordii in their repositories. You can install it using your system's package manager.

Distributions that have Chordii are a.o. Fedora, Debian and Ubuntu.

From the SourceForge web site

On the Chordii project page on SourceForge, <http://www.sourceforge.net/projects/chordii/>, you can find binary distributions of Chordii for popular systems as Linux, Windows and Mac OS/X.

Building from source

Building from source requires a C compiler, and, preferably, the GNU Build System. This is commonly available on most modern computers running the GNU/Linux and Mac OS/X operating systems.

You can find the Chordii sources and anything else you need on the Chordii project page on SourceForge, <http://www.sourceforge.net/projects/chordii/>.

See the INSTALL file in the top level source directory for build and installation instructions for Linux, Mac/OSX and most other Unix based systems.

For RPM based systems, building your own RPM is as easy as:

```
$ rpmbuild -ta chordii-4.5.3.0.tar.gz
```

For Debian systems, you can build a DEB distribution for Chordii as follows:

```
$ tar -zxf chordii-4.5.3.0.tar.gz
$ cd chordii-4.5.3.0
$ fakeroot dpkg-buildpackage -b
```

For Windows systems, unpack the source kit, and then:

```
C> cd chordii-4.5.3.0\windows
C> nmake
```

Note that in the above instructions, the version number must be changed when new releases are made.

Appendix B. Where to find Chordii files

Chordii's predecessor has been around on the internet since 1991 so there are many song files available on the Internet. Also, several other programs exist that use the same, or almost the same, data format. Their input files may be suitable for Chordii as well.

Internet search engines can help you to locate songs and lyrics for a great many pop and traditional songs. Common filename extensions for files suitable for Chordii are `crd`, `cho`, `crdpro`, `chopro`, and `chordpro`.

Appendix C. Some Future plans

PDF generation

Instead of PostScript, produce PDF. Natively? Or with a wrapper script that invokes Ghostscript?

UNICODE

Allow input in UTF-8 and UTF-16. How to deal with that in the generated PostScript?