Table of Contents

I. About the Package 2
   1. License and Requirements 2
   2. Background 2
   3. The Structure of the Package 3

II. The musicsymbols Library 5

III. The chords Library (and the musejazz Library) 7
   4. The writechord Command 7
   5. Options 9
   6. The musejazz Library 11

IV. The songs Library 13
   7. Background 13
   8. The song Environment 13
      8.1. A First Example 14
      8.2. Using the song Environment 15
      8.3. Options 16

     9. The verse Environment 19
        9.1. Options 21

     10. Placing Chords 22
        10.1. The Commands 22
        10.2. Usage 22
        10.3. Caveat 24
        10.4. Remembering Chords 25

     11. Transposing 27

     12. Other verse-like Environments 29
        12.1. Available Environments 29
        12.2. Own verse-like Environments 30

     13. Typesetting Bars 32

     14. Templates 33
        14.1. Title Templates 33
        14.1.1. Background 33
        14.1.2. Existing Templates 33
        14.1.3. Own Templates 33
        14.1.4. Examples 35

        14.2. Verse-type Templates 39
        14.2.1. Background 39
1. License and Requirements

Permission is granted to copy, distribute and/or modify this software under the terms of the \LaTeX{} Project Public License (\LPPL{}), version 1.3c or later (http://www.latex-project.org/lppl.txt). The software has the status “maintained.”

\textsc{leadsheets} requires the bundles \texttt{l3kernel} [L3Pa] and \texttt{l3packages} [L3Pb] to be available. It also needs the package \textsc{translations} [Nie20].

2. Background

Over the years I repeatedly wanted to typeset simple leadsheets of songs, \textit{i.e.}, song lyrics and the corresponding chords.\textsuperscript{1} This is not too hard with standard \LaTeX{} commands and environments but it is not very convenient. So looking for existing packages is the logical next step and I found two very promising packages: songs [Ham18] and songbook [Rat10]. However, both were not quite what I wanted. Just a bit too inflexible in the wrong places, needing tweaking here and there, and so on. On the other hand I had quite some code lying on my hard drive with various attempts of typesetting leadsheets. This package is now the attempt to have a clean, documented and customizable version of my code.\textsuperscript{2}

\begin{flushright}
I like beautiful melodies telling me terrible things. \\
\textit{Tom Waits}
\end{flushright}

\textsuperscript{1} I also have had the need (or let’s say: wish) to typeset leadsheets in the style of the \textit{The Real Book} – but this is a task where other software than \LaTeX{} usually is far easier.

\textsuperscript{2} Plus new things!
3. The Structure of the Package

**LEADSHEETS** is a modular package and consists of several libraries. If you just say

```latex
\usepackage{leadsheets}
```

then the **songs** library is loaded. If you want to ensure that every library is loaded so you don’t have to think about it any more you can use

```latex
\usepackage[full]{leadsheets}
```

This actually loads every library except the **musejazz** library.

If you instead use

```latex
\usepackage[minimal]{leadsheets}
```

then no libraries are loaded. In this case you have to load the libraries yourself in order to use the package.

```latex
\useleadsheetslibraries{(comma list of libraries)}
```

With this command one or several of **LEADSHEETS**’ libraries can be loaded.

```latex
\useleadsheetslibrary{(library)}
```

With this command one of **LEADSHEETS**’ libraries can be loaded.

The libraries are divided into two parts: libraries to be loaded by users and auxiliary libraries loaded by other libraries but not to be loaded directly by users.

The user-libraries are the following ones:

- **musicsymbols**  This library makes the music symbol font provided by MusiXTeX available as text font. It is described in part II.
- **chords**   This library defines a few macros for typesetting of chord symbols. It is described in part III. It also loads the **musicsymbols** library.

3. This did seem like a good idea for maintenance when I first came up with it. Now I’m not so sure any more but I am way too lazy to change it back. So here we are...
4. Which currently means that all libraries are loaded as the **songs** library needs them.
3. The Structure of the Package

**Package option: musejazz**  
*musejazz* extends the *chords* library to use MuseScore’s “MuseJazz Text” font, see section 6. It loads the *chords* library.

**Package option: songs**  
*songs* This is the *main library*. It defines everything necessary for the typesetting of the leadsheets. It currently loads all other libraries, i.e., user libraries and internal libraries, except for the *musejazz* library. This library is described in part IV.

**Package option: external**  
*external* enables to include external leadsheet documents using the *leadsheets* class into a main document, see part V for details.

Those libraries can also be loaded as a package option. If you say

```latex
\usepackage[musicsymbols]{leadsheets}
```

then *only* the *musicsymbols* library is loaded. Saying

```latex
\usepackage[songs]{leadsheets}
```

is the same as using the package *without* any options. “Without any” also means that neither *musicsymbols*, *chords* nor *songs* has been given as class option!

There is also a number of auxiliary libraries which are all needed by the *songs* library. The libraries are usually not described in sections of their own but as part of the *songs* library. However, the following list gives hints to where some of the concepts are explained:

**properties**  
This is an auxiliary library not meant to be loaded directly by users. It defines the necessary macros for song properties. See section 8.4 for more details on the concept.

**shorthands**  
This is an auxiliary library not meant to be loaded directly by users. It provides means for defining shorthand characters.

**transposing**  
This is an auxiliary library not meant to be loaded directly by users. It defines a transposing mechanism for chord symbols. See section 11 for further details.

**chordnames**  
This is an auxiliary library not meant to be loaded directly by users. It defines the necessary macros for printing chords.

**templates**  
This is an auxiliary library not meant to be loaded directly by users. It defines the necessary macros for *leadsheets*’ template mechanism. See section 14 for details on the concept.

**translations**  
This is an auxiliary library not meant to be loaded directly by users. It provides a few translations for a number of printed strings. See section 15 for more information.
The **musicsymbols** library is a very small library that makes the music symbol font provided by MusiXTeX available as text font and then uses it to define a number of symbols. This redefines the macros \texttt{\textbackslash sharp}, \texttt{\textbackslash flat} and \texttt{\textbackslash natural}. All defined symbols are listed in table 1.

If you want to use the library standalone then you can say:

\begin{verbatim}
1 \usepackage[minimal]{leadsheets}
2 \useleadsheetslibraries{musicsymbols}
\end{verbatim}

**musicsymbols** defines three further commands, namely \texttt{\textbackslash musix}, \texttt{\textbackslash textmusix} – a font switch and a text font command – and \texttt{\textbackslash musicsymbol}. Those commands are meant for internal use only which is why they’re not explained here.

<table>
<thead>
<tr>
<th>Command</th>
<th>Symbol</th>
<th>Command</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>\texttt{\textbackslash sharp}</td>
<td>#</td>
<td>\texttt{\textbackslash flat}</td>
<td>\flat</td>
</tr>
<tr>
<td>\texttt{\textbackslash doublesharp}</td>
<td>#</td>
<td>\texttt{\textbackslash doubleflat}</td>
<td>\flat</td>
</tr>
<tr>
<td>\texttt{\textbackslash natural}</td>
<td>\natural</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\texttt{\textbackslash trebleclef}</td>
<td>\trebleclef</td>
<td>\texttt{\textbackslash bassclef}</td>
<td>\bassclef</td>
</tr>
<tr>
<td>\texttt{\textbackslash altoclef}</td>
<td>\altoclef</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\texttt{\textbackslash allabreve}</td>
<td>\allabreve</td>
<td>\texttt{\textbackslash meterC}</td>
<td>\meterC</td>
</tr>
<tr>
<td>\texttt{\textbackslash wholesrest}</td>
<td>\wholerest</td>
<td>\texttt{\textbackslash halfrest}</td>
<td>\halfrest</td>
</tr>
<tr>
<td>\texttt{\textbackslash quarterrest}</td>
<td>\quarterrest</td>
<td>\texttt{\textbackslash eighrestart}</td>
<td>\eighrestart</td>
</tr>
<tr>
<td>\texttt{\textbackslash sixteenthrest}</td>
<td>\sixteenthrest</td>
<td></td>
<td></td>
</tr>
<tr>
<td>\texttt{\textbackslash Break}</td>
<td>\Break</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**musicsymbols** also defines a number of macros for denoting bars. Those macros are listed in table 2.
<table>
<thead>
<tr>
<th>Macro</th>
<th>Symbol</th>
<th>Macro</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>normalbar</td>
<td>\normalbar</td>
<td>leftrepeat</td>
<td>\leftrepeat</td>
</tr>
<tr>
<td>rightrepeat</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>doublebar</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

There are three macros that can be redefined to adjust the appearance of those bars:

\normalbarwidth
   The thickness of a bar rule as used in \normalbar.

\thickbarwidth
   The thickness of the thick bar rules as used for example in \leftrepeat or \stopbar.

\interbarwidth
   The distance between two bar rules as in \doublebar or \stopbar.

Finally musicsymbols provides a macro for adding an arbitrary meter:

\meter{⟨numerator⟩}{⟨denominator⟩}

For example \meter{6}{8}: \frac{6}{8}
Part III.
The chords Library (and the musejazz Library)

I never practice my guitar – from time to time I just open the case and throw in a piece of raw meat.

Wes Montgomery

4. The \writechord Command

chords provides the command \writechord{⟨chord⟩} for convenient typesetting of chords:

\[ \text{\writechord{Bb7(#9)} \writechord{Bbb6}} \]
\[ \text{\writechord{C#7(b9)} \writechord{C##13}} \]
\[ \text{\chordname{⟨chord⟩}} \]

Typesetting chords. Inside the argument every # will be replaced by # and every b is replaced with b. Numerals and parentheses are typest as superscripts. Everything between parentheses is always a superscript: \writechord{F#7(#11)} F#7(#11).

\text{\writechord{⟨chord⟩}}

This command can and actually should be used for placing chords inline. It transforms the chords according to the options knows the options input-notation and output-notation. This command also transforms the chord according to the options inside of the song environment transpose and enharmonic before printing it, see also page 22.

There are several token lists that are treated specially inside \writechord:

\[ ^* \]

- This token is replaced by \textsuperscript.
4. The \texttt{writechord} Command

- \texttt{ma} – The symbol for major chords. Per default this is empty. \texttt{writechord(Gma)} G.

- \texttt{mi} – The symbol for minor chords. Per default this is m. \texttt{writechord(Gmi)} Gm.

- \texttt{o} – The symbol for diminished chords. Per default this is \textsuperscript{o}. \texttt{writechord(Go)} Go.

- \texttt{+} – The symbol for augmented chords. Per default this is \textsuperscript{+}. \texttt{writechord(G+)} G+.

- \texttt{/o} – The symbol for half diminished chords. Per default this is \textsuperscript{o}. \texttt{writechord(G/o)} Gø.

- \texttt{#} – The “sharp” symbol. Per default this is \texttt{sharp}. \texttt{writechord(F#)} F#.

- \texttt{##} – The “double sharp” symbol. Per default this is \texttt{doubleshap}. \texttt{writechord(F##)} Fx.

- \texttt{b} – The “flat” symbol. Per default this is \texttt{flat}. \texttt{writechord(Eb)} Eb.

- \texttt{bb} – The “double flat” symbol. Per default this is \texttt{doubleflat}. \texttt{writechord(Ebb)} Eb.

- \texttt{b#} – Cancelling flat/sharp combination: this is removed.

- \texttt{#b} – Cancelling sharp/flat combination: this is removed.

- \texttt{add} – This is superscripted: \texttt{writechord(Gadd9)} G\textsuperscript{add9}.

- \texttt{sus} – This is superscripted: \texttt{writechord(Gsus4)} G\textsuperscript{sus4}.

- \texttt{dim} – This is superscripted: \texttt{writechord(Gdim5)} G\textsuperscript{dim5}.

- \texttt{maj7} – This is superscripted: \texttt{writechord(Gmaj7)} G\textsuperscript{maj7}.

- \texttt{maj9} – This is superscripted: \texttt{writechord(Gmaj9)} G\textsuperscript{maj9}.

How these token lists are treated depends on optional settings:

\begin{verbatim}
\setchords{
  major-seven = $\Delta$
  major-nine = $\Delta^9$
}
\end{verbatim}

\begin{verbatim}
\writechord{Gmaj7} \writechord{Gmaj9}
\writechord{G6\#5} \writechord{G6}
\writechord{G7\#5}
\end{verbatim}

GΔ GΔ\textsuperscript{9} G\textsuperscript{6} G\textsuperscript{6} G\textsuperscript{7\#5}
5. Options

If you want to use the library standalone then you can say:

1. \usepackage[minimal]{leadsheets}
2. \useleadsheetslibraries{chords}

This also loads the musicsymbols library.

5. Options

Options are set with the command

\setchords{⟨options⟩}

where ⟨options⟩ is a comma separated list of keyval options.

Actually there’s a second possibility: options can also be set with the command \setleadsheets (see section 8.3) if they’re preceded by chords/ (including the slash).

The options allow detailed customization of how chords are printed. It doesn’t change the input syntax.

format = ⟨⟨code⟩⟩  (initially empty)

Code inserted before a chord within the same group. Can be used for special formatting of the chords, with \sffamily, say.

sharp = ⟨⟨code⟩⟩  Default: \sharp

The sharp symbol.

flat = ⟨⟨code⟩⟩  Default: \flat

The flat symbol.

double-sharp = ⟨⟨code⟩⟩  Default: \doublesharp

The double sharp symbol.

double-flat = ⟨⟨code⟩⟩  Default: \doubleflat

The double flat symbol.

aug = ⟨⟨code⟩⟩  Default: +

The augmented symbol.

half-dim = ⟨⟨code⟩⟩  Default: \o{}

The half-diminished symbol.

full-dim = ⟨⟨code⟩⟩  Default:  o

The diminished symbol.
5. Options

\texttt{dim} = \{\langle code \rangle\}

The token list \texttt{dim}. Default: \textsuperscript{dim}

\texttt{add} = \{\langle code \rangle\}

The token list \texttt{add}. Default: \textsuperscript{add}

\texttt{sus} = \{\langle code \rangle\}

The token list \texttt{sus}. Default: \textsuperscript{sus}

\texttt{major} = \{\langle code \rangle\}

The token list \texttt{major}. (initially empty)

\texttt{minor} = \{\langle code \rangle\}

The token list \texttt{minor}. This option knows a special value. If you set \texttt{minor} = \{\texttt{lowercase}\} then minor chords are converted to lowercase letters. This means an input \texttt{Dmi} becomes \texttt{d}.

\texttt{major-seven} = \{\langle code \rangle\}

The token list \texttt{maj7}. Default: \textsuperscript{maj7}

\texttt{major-nine} = \{\langle code \rangle\}

The token list \texttt{maj9}. Default: \textsuperscript{maj9}

There are further options which concern the different notations of the pitches \texttt{B} and \texttt{B}\texttt{b} in German (and a few other European countries) and English.

\texttt{input-notation} = \texttt{german}|\texttt{english}

If set to \texttt{german} \texttt{B} should be input as \texttt{H} and \texttt{B flat} as \texttt{B}. If set to \texttt{english} \texttt{B} should be input as \texttt{B} and \texttt{B flat} as \texttt{Bb}.

\texttt{output-notation} = \texttt{german}|\texttt{english}

If set to \texttt{german} \texttt{B} is output as \texttt{H} and \texttt{B-flat} as \texttt{B}.

\texttt{german-B} = \{\langle code \rangle\}

Customize the output of \texttt{B flat} when \texttt{output-notation} = \{\texttt{german}\}. Default: \texttt{B}

\texttt{german-H} = \{\langle code \rangle\}

Customize the output of \texttt{B} when \texttt{output-notation} = \{\texttt{german}\}. Default: \texttt{H}

Let's see a few examples for the \texttt{input-notation} and \texttt{output-notation} options.

\begin{verbatim}
1 input and output English:\par
2 writechord(Bb) writechord(B) writechord(B#)\par
3 writechord(B) writechord(H) writechord(H#)\par
4 medskip
5\setchords{input-notation=german}
6 input German and output English:\par
7 writechord(Bb) writechord(B) writechord(B#)\par
\end{verbatim}
The musejazz Library

The musejazz library extends the chords library to use MuseScore’s font “MuseJazz Text” for the chord symbols. The library contains the two lines which explains why it requires LuaLaTeX or XeLaTeX. Well, and the font, obviously.

\begin{Verbatim}
\RequirePackage{fontspec}
\newfontfamily\musejazz{MuseJazz Text}
\end{Verbatim}

Here is a small example of its usage and the outcome:

\begin{Verbatim}
\writechord{B} \writechord{H} \writechord{H#} \par
\medskip
\setchords{output-notation=german}
\writechord{Bb} \writechord{B} \writechord{B#} \par
\writechord{B} \writechord{H} \writechord{H#} \par
\medskip
\setchords{input-notation=english}
\input and output English:
B♭ B B♯
B B B♯
input German and output English:
B♭♭ B♭ B
B♭ B B♯
input and output German:
B♭ B H
B H H♯
input English and output German:
B H H♯
H H H♯
\end{Verbatim}
6. The `musejazz` Library

```latex
% compile with LuaLaTeX or XeLaTeX
\documentclass[margin=4mm]{standalone}
\usepackage{leadsheets}
\useleadsheetslibrary{musejazz}
\begin{document}
\Huge\writechord{Bb7(#9)} \space \writechord{F##9}
\end{document}
```
Part IV.
The songs Library

7. Background

The leadsheets package allows for easy creation of leadsheets but it also can be used to create complete songbooks. The basic idea is that songs are typeset in the song environment. Each song gets a number of properties (see section 8.4) that determine how the title of the song is typeset. For the typesetting of the titles a template mechanism is used (see section 14.1). Songs can also be tagged. The tags then allow to typeset only songs matching a list of tags that is specified via an option.

8. The song Environment

\begin{song}\{\langle options \rangle \}\{\langle properties \rangle \}\end{song}

The main environment used to typeset songs. It has a mandatory argument where the song’s
properties are set (cf. section 8.4). It also has an optional argument for setting options specific to the song.

8.1. A First Example

First let’s take a look at an example:

```latex
\documentclass{article}
\usepackage{leadsheets}
\begin{document}
\begin{song}{title={Mary Had A Little Lamb}, music={Stevie Ray Vaughan},
lyrics={traditional}, tags={srv,blues}}
\begin{verse}
Mary had a little lamb \\
Its fleece was white as snow, yea \\
Everywhere the child went \\
The little lamb was sure to go, yea
\end{verse}
\begin{verse}
He followed her to school one day \\
And broke the teacher’s rule \\
What a time did they have \\
That day at school
\end{verse}
\end{song}
\end{document}
```
8. The song Environment

Mary Had A Little Lamb

Mary had a little lamb
Its fleece was white as snow, yea
Everywhere the child went
The little lamb was sure to go, yea

He followed her to school one day
And broke the teacher’s rule
What a time did they have
That day at school

Per default the songtitle simply is a \section* without any other additions. This is the songtitle template minimal, see section 14.1 for more details on those templates and how to create your own.

8.2. Using the song Environment

Inside the song environment a number of additional environments are used to specify the different parts of a song. They all are basically the same kind of environment, namely an itemize environment internally where the only \item has the name of the environment as option. The verse environment is a little bit different since verses can be numbered. If they are then each usage of verse inside song will step a verse number and print it (as option to the internal \item).

\begin{verse}[⟨options⟩]
An environment for specifying the verses of a song.
\end{verse}

\begin{chorus}[⟨options⟩]
An environment for specifying the chorus of a song.
This is the same as \begin{verse}[type=chorus,⟨options⟩].
\end{verse}

\begin{intro}[⟨options⟩]
An environment for specifying the intro of a song.
This is the same as \begin{verse}[type=intro,⟨options⟩].
\end{verse}

\begin{interlude}[⟨options⟩]
An environment for specifying an interlude of a song.
This is the same as \begin{verse}[type=interlude,⟨options⟩].
\end{verse}
8. The song Environment

\begin{bridge}\[\_bridge\] \begin{verse}\[\type=bridge,\langle\_options\_\rangle\] An environment for specifying a bridge of a song. This is the same as \begin{verse}\[\type=bridge,\langle\_options\_\rangle\]. \end{verse} \end{verse} These environments and their options are described in more detail in sections 9 and 12.

8.3. Options

The options to the song environment are the same as the general options of leadsheets. This means you can set the following options either local to a song or global for the whole document with this command:

\setleadsheets\[\langle\_options\_\rangle\] Setup command for leadsheets.

Although I used the word “global” above all options are local to the current scope!

\begin{quote}
\textbf{title-template} = \{\langle\_template\_name\_\rangle\} Default: minimal
\end{quote}

The song title template, see section 14.1 for details.

\begin{quote}
\textbf{chord-cs} = \{\langle\_cs\_\rangle\} Default: \chordname
\end{quote}

The command that is used to parse the chords. See section 10 for details. \langle cs \rangle needs to be a command that takes a mandatory argument.

\begin{quote}
\textbf{song-format} = \{\langle\_code\_\rangle\} (initially empty)
\end{quote}

\langle code \rangle is inserted before the song title at the beginning of the song environment.

\begin{quote}
\textbf{text-format} = \{\langle\_code\_\rangle\} (initially empty)
\end{quote}

\langle code \rangle is inserted after the song title at the beginning of the song environment.

\begin{quote}
\textbf{before-song} = \{\langle\_code\_\rangle\} (initially empty)
\end{quote}

\langle code \rangle is inserted at the beginning of a song before the title!

\begin{quote}
\textbf{after-song} = \{\langle\_code\_\rangle\} (initially empty)
\end{quote}

\langle code \rangle is inserted after the whole song.

\begin{quote}
\textbf{after-title} = \{\langle\_code\_\rangle\} (initially empty)
\end{quote}

\langle code \rangle is inserted after the title of the song but before the body of the environment.

\begin{quote}
\textbf{print-tags} = \{\langle\_list\_of\_tags\_\rangle\}
A comma separated list of tags. When specified a song will only be printed if it is tagged with at least one of the tags in \langle list of tags \rangle.
\end{quote}

\begin{quote}
\textbf{add-to-reset} = \{\langle\_list\_of\_counters\_\rangle\}
A comma separated list of counters which should be reset for each song.
\end{quote}
8. The song Environment

**obey-lines** = true|false  Default: false
An experimental option. Use at your own risk! If set to true then inside the verse like environments end-of-lines will be obeyed and start a new line. This comes with a price when using chords, see section 10.3.

**bar-shortcuts** = true|false  Default: false
Makes the characters : and | active inside the song environment. See sections 8.5 and 13 for more details.

**disable-measuring** = true|false  Default: false
This option turns off the song height measuring. This makes song title template definitions easier (see section 14.1 for details). The song property height will be set to opt then!

### 8.4. Song Properties

Songs can have a number of properties which basically are used in songtitle templates (see section 14.1). One specific property, tags, plays a different role, though.

**title** = {
  ⟨title⟩
}
This is the main title of the song.

**subtitle** = {
  ⟨subtitle⟩
}
A subtitle.

**short-title** = {
  ⟨short song title⟩
}
A short title (may be useful in a template that writes the titles in \sections for a version to be used in the table of contents).

**sort-title** = {
  ⟨song title⟩
}
If not set explicitly this property holds the same value as title.

**sort-short-title** = {
  ⟨short song title⟩
}
If not set explicitly this property holds the same value as short-title.

**composer** = {
  ⟨composer⟩
}
The composer of the song. As of now this accepts an arbitrary entry but maybe this will not be supported any more when indexing will be implemented. No promises.

**sort-composer** = {
  ⟨composer⟩
}
If not set explicitly this property holds the same value as composer.

**lyrics** = {
  ⟨writer⟩
}
Whoever wrote the lyrics if different from the composer. As of now this accepts an arbitrary entry but maybe this will not be supported any more when indexing will be implemented. No promises.

**sort-lyrics** = {
  ⟨writer⟩
}
If not set explicitly this property holds the same value as writer.
8. The song Environment

\[ \text{arr} = \{\text{arranger}\} \]
Whoever arranged the song. As of now this accepts an arbitrary entry but maybe this will not be supported any more when indexing will be implemented. No promises.

\[ \text{sort-arr} = \{\text{arranger}\} \]
If not set explicitly this property holds the same value as \text{arr}.

\[ \text{band} = \{\text{band}\} \]
The band who plays or played the song.

\[ \text{sort-band} = \{\text{band}\} \]
If not set explicitly this property holds the same value as \text{band}.

\[ \text{interpret} = \{\text{interpret}\} \]
The interpret of the song. As of now this accepts an arbitrary entry but maybe this will not be supported any more when indexing will be implemented. No promises.

\[ \text{sort-interpret} = \{\text{interpret}\} \]
If not set explicitly this property holds the same value as \text{interpret}.

\[ \text{genre} = \{\text{genre}\} \]
The genre of the song.

\[ \text{key} = \{\text{key}\} \]
The key of the song. This property is used for transposing and must have a specific format then, see section 11.

\[ \text{capo} = \{\text{fret}\} \]
This property is used for transposing and for the \texttt{\capo} macro, see sections 8.6 and 11.

\[ \text{tempo} = \{\text{tempo}\} \]
The tempo of the song.

\[ \text{tags} = \{\text{tags}\} \]
A comma separated list of tags. Those tags play a role for the option \texttt{print-tags}. When that option is used a song is only printed if it has at least one of the tags specified in the option.

There are three more properties, \texttt{counter}, \texttt{ID} and \texttt{height} that cannot be set but are set automatically for each song. The \texttt{counter} simply holds the number of the current song starting from 1 for the first song. The \texttt{ID} currently always is \texttt{song}\{\texttt{counter}\} where \texttt{\{\texttt{counter}\}} is the current \texttt{counter} value. The property \texttt{height} holds the height of the typeset song in pt. The height is determined by placing the body of the respective \texttt{song} environment in a vertical box and measuring the height and depth of the box. This is done in a measuring phase that can and should be tested in a songtitle template definition, see section 14.1 for details. \textit{This is important since the property height is not available in the measuring phase but only afterwards!} If the option \texttt{disable-measuring} has been set the property \texttt{height} will have the value opt!

In principle all properties can get list of entries where items are separated with \texttt{\_and\_}. Of course this doesn’t make sense for each property – a song does only have one title. But a
song can very well have more than one composer: think of the Beatles where most songs were written by Paul McCartney and John Lennon.\footnote{This is not quite true: most songs were mostly written either by Paul or John but legally usually both are the composers.}

It is possibly to define further such properties. For details see section 14.1.3.

\section{Special Characters}

Inside the \texttt{song} environment several characters don’t have their usual category codes:

\begin{itemize}
\item ^ – category code 13 (active). It is a shortcut for the \texttt{chord} command.
\item _ – category code 13 (active). It is a shortcut for the \texttt{writechord} command.
\item | – category code 13 (active). Used for typesetting bars.
\item : – category code 13 (active). Used for typesetting bars.
\item # – category code 12 (other). Used for chord names.
\end{itemize}

Actually the characters | and : are \textit{not} changed per default. In order to do that you have to use the option \texttt{bar-shortcuts}.

For details on the usage of the characters | and : see section 13. The usage of chords is explained in section 10.

\section{Capo Information}

When you set the \texttt{capo} property the macro \texttt{\capo} writes

\begin{verbatim}
capo: IV. fret
\end{verbatim}

What it writes exactly depends on a few settings: the \texttt{capo} property obviously, which determines the number that is printed. The translations for the "capo" and "fret" strings (see section 15 for details) and the setting of the following option:

\begin{itemize}
\item \texttt{capo-nr-format = arabic|roman}\texttt{Roman} \hfill \texttt{Default: Roman}
\item \texttt{capo-nr = \{⟨code⟩\}} \hfill \texttt{Default: \#1.}
\end{itemize}

\begin{itemize}
\item The format of the number printed by the \texttt{\capo} macro.
\item The code to print the number. In the code refer to the number with \#1.
\end{itemize}

\section{The verse Environment}

\begin{verbatim}
\begin{verse}\{⟨options⟩\}
\end{verse}
\end{verbatim}

An environment for specifying the verses of a song.

\begin{verbatim}
\begin{verse*}\{⟨options⟩\}
\end{verse*}
\end{verbatim}

The same as the \texttt{verse} environment but will always be unnumbered regardless of any option settings.

\footnote{This is not quite true: most songs were mostly written either by Paul or John but legally usually both are the composers.}
9. The verse Environment

\documentclass{article}
\usepackage{leadsheets}
\setleadsheets{verse/numbered=true}
\begin{document}
\begin{song}{title=Foo}
\begin{verse}
Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\
sed diam nonumy eirmod tempor invidunt ut labore et dolore\
magna aliquyam erat, sed diam voluptua.
\end{verse}
\begin{verse*}
Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\
sed diam nonumy eirmod tempor invidunt ut labore et dolore\
magna aliquyam erat, sed diam voluptua.
\end{verse*}
\begin{verse}
Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\
sed diam nonumy eirmod tempor invidunt ut labore et dolore\
magna aliquyam erat, sed diam voluptua.
\end{verse}
\end{song}
\end{document}

Foo

1. Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
   sed diam nonumy eirmod tempor invidunt ut labore et dolore
   magna aliquyam erat, sed diam voluptua.
   
   Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
   sed diam nonumy eirmod tempor invidunt ut labore et dolore
   magna aliquyam erat, sed diam voluptua.

2. Lorem ipsum dolor sit amet, consetetur sadipscing elitr,
   sed diam nonumy eirmod tempor invidunt ut labore et dolore
   magna aliquyam erat, sed diam voluptua.
9. The verse Environment

9.1. Options

The verse environment and all related environments have the following options:

- **format** = {⟨code⟩}          (initially empty)
  ⟨code⟩ is inserted at the beginning of the environment and can thus be used to add formatting, e.g., format = {\itshape}.

- **label-format** = {⟨code⟩}     (initially empty)
  The same for the environment labels. The last command may take the label as an argument so you can also use \textbf for example.

- **class** = {⟨class-name⟩}      Default: default
  The verse environment and all related environments all belong to a class, the default class is called default. This is of interest when using the remember-chords and recall-chords options.

This can be used either locally, i.e., as option to the corresponding environment, or for all environments of the same type using the setup command using the following syntax:

\setleadsheets{⟨env name⟩/format = ⟨code⟩}

```
\begin{verse}[format=\itshape]
Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\
sed diam nonumy eirmod tempor invidunt ut labore et dolore\
magna aliquyam erat, sed diam voluptua.
\end{verse}
```

It is also possible so set the formatting for all related environments at once:

- **verses-format** = {⟨code⟩}    (initially empty)
  Sets the format for all verse like environments.

- **verses-label-format** = {⟨code⟩} (initially empty)
  Sets the label format for all verse like environments.

Both options are overwritten if the options for a specific environment are set. That is, if you want all environments to have italic shape except for choruses, then you could do
10. Placing Chords

10.1. The Commands

Inside the \texttt{song} environment the characters ^ and _ are active characters. ^ is a shortcut for the command \texttt{\chord}, _ is a shortcut for \texttt{\writechord}. Those commands have the following functions:

\texttt{\chord*-\{\langle chord \rangle \langle text \rangle \}}

Places \langle chord \rangle centered above \langle text \rangle. The starred version gobbles the trailing space while the un-starred version does not. Like the star the dash is optional. It sets the option \texttt{smash-next-chord} to true. \langle text \rangle may be empty but the trailing space must be there. If \langle text \rangle is empty then the chord is place above some horizontal space which can be set with the option \texttt{empty-chord-dim}.

\texttt{\writechord\{\langle chord \rangle\}}

This command transforms the chord according to the options \texttt{transpose} and \texttt{enharmonic} before printing it. This command can/should be used for placing chords inline or for typesetting the \texttt{key} property in a template. The command is used by \texttt{\chord} for the actual printing. The command also transforms the chords according to the options \texttt{input-notation} and \texttt{output-notation}.

\texttt{Text \chord{E7} \chord*{B7} longer text}

E\textsuperscript{7} B\textsuperscript{7}

Text text longer text

10.2. Usage

Note that per default the width of a chord is not ignored:

\texttt{text \chord{Gbmi7(b5)} text}

G\textsuperscript{bmi7(b5)}

text text text

However, there is an option which sets the width of a chord to zero:

7. There are more characters with a special function, see section 8.5
10. Placing Chords

`smash-chords = true|false`  
*Default: false*  
If set to true the width of the chords set with \chord is set to zero.

`smash-next-chord = true|false`  
*Default: false*  
If set to true the width of the next chord set with \chord is set to zero.

\begin{verbatim}
\setleadsheets{smash-next-chord=true}
\chord{Gbmi7(b5)} text text text
\chord{Gbmi7(b5)} text text \par
\setleadsheets{smash-chords=true}
\chord{Gbmi7(b5)} text text \par
\chord{Gbmi7(b5)} text text
\end{verbatim}

Before we forget – there are more options:

`empty-chord-dim = \langle dim \rangle`  
*Default: 1em*  
The horizontal space that is inserted if the \langle text \rangle argument of \chord is empty.

`align-chords = \langle col \rangle`  
*Default: c*  
Determines how a chord is aligned with respect to the word it is placed above of. Valid input is any valid tabular column identifier.

`print-chords = true|false`  
*Default: true*  
If set to false \chord won’t print the chord but will do nothing instead. This options does not affect \writechord.

While \chord is available in the whole document the \^ syntax is – as mentioned before – only available inside of the song environment.

\begin{verbatim}
\documentclass{article}
\usepackage{leadsheets}
\begin{document}
\begin{song}{title={Mary Had A Little Lamb}, music={Stevie Ray Vaughan},
lyrics={traditional}, tags={srv,blues}}
\begin{verse}
^{E7} Mary had a little lamb ^{A7} \\
Its fleece was white as ^{E7}snow, yea \\
Everywhere the child ^{B7}went ^{A7} \\
The little lamb was sure to ^{E7}go, yea
\end{verse}
\end{song}
\end{document}
\end{verbatim}
10. Placing Chords

You’ve probably noticed: chords are printed with \chordname{} in the default setting. You can change this with the option chord-cs. If you do then keep in mind that the input syntax will also change.

10.3. Caveat

If you use obey-lines = {true} you have to be careful when you place chords. If you place a chord over the last word in a line

\begin{verse}
^{F#mi} You’ve been ^{*}(B)run nin’, ^{*}(E)hid in’ much too ^{A}long.
\end{verse}

then the end of line that is used as the mandatory space argument for \chord{} may not be recognized as an end of line. Even worse: at the end of a verse like environment this may cause
non-obvious errors. So if you’re getting in trouble in these cases you should always insert an explicit space, e.g. by one of the following methods:

```latex
1. ^{F#mi} You’ve been ^{B}run nin’, ^{E}hid in’ much too ^{A}long. {}
2. ^{F#mi} You’ve been ^{B}run nin’, ^{E}hid in’ much too ^{A}long. \empty
3. ^{F#mi} You’ve been ^{B}run nin’, ^{E}hid in’ much too ^{A}long. \relax
```

## 10.4. Remembering Chords

**Leadsheets** has the option

```latex
\texttt{remember-chords = true|false}
```

Default: false

If set to true the chords in the first appearance of a verse like environment are remembered. In the next appearances of said environment the shortcut `^` has changed its meaning and inserts a chord automatically.

Let’s take at look at an example to see what this means:

```latex
\definesongtitletemplate{empty}{}
\begin{song}{verse/numbered,remember-chords,title-template=empty}{title=foobar}
\begin{verse}
^{G}Lorem ipsum ^{C}dolor sit ^{E7}amet, consetetur ^{Bb7(b5)}sadipscing
\end{verse}
\begin{verse}
^{Lorem ipsum ^dolor sit ^amet, consetetur ^sadipscing
\end{verse}
\end{song}
```

1. G Lorem ipsum C dolor sit E7 amet, consetetur Bb7(b5)sadipscing
2. G Lorem ipsum dolor sit amet, consetetur sadipscing

In this example the chords used in the first verse environment have been remembered and in the second verse environment the `^` shortcut inserted the corresponding chords in the order they had been specified the first time. It is important when using this that you don’t use the `^` shortcut in subsequent environments more often than the first time. It will produce an error otherwise. You can use it less, of course.
10. Placing Chords

The ^ shortcut still has the *same* syntax as \chord with one exception: it lacks the mandatory argument ⟨chord⟩ (since it's obviously not needed). It has the optional * and -, though, as well as the mandatory space ()!

*Please note that ^ can’t be the last token in a line! It then should be followed by a space an an empty group – similar to the caveat in section 10.3.

Each verse like environment (see section 12 for more information) is treated uniquely by this mechanism:

```
\definesongtitletemplate{empty}{}
\begin{song}[verse/numbered,remember-chords,title-template=empty]{title=foobar}
\begin{verse}
^{G}Lorem ipsum ^{C}dolor sit ^{E7}amet, consetetur ^{Bb7(b5)}sadipscing
\end{verse}
\begin{chorus}
^{F}Lorem ipsum ^{Gmi}dolor sit amet, consetetur ^{C7}sadipscing
\end{chorus}
\begin{verse}
^Lorem ipsum ^dolor sit ^amet, consetetur ^sadipscing
\end{verse}
\begin{chorus}
^Lorem ipsum ^dolor sit amet, consetetur ^sadipscing
\end{chorus}
\end{song}
```

\[
\begin{array}{cccc}
G & C & E^7 & B_b^7(b5) \\
1. Lorem ipsum dolor sit amet, consetetur sadipscing \\
\end{array}
\]

\[
\begin{array}{cccc}
F & Gm & C^7 \\
Chorus: Lorem ipsum dolor sit amet, consetetur sadipscing \\
\end{array}
\]

\[
\begin{array}{cccc}
G & C & E^7 & B_b^7(b5) \\
2. Lorem ipsum dolor sit amet, consetetur sadipscing \\
\end{array}
\]

\[
\begin{array}{cccc}
F & Gm & C^7 \\
Chorus: Lorem ipsum dolor sit amet, consetetur sadipscing \\
\end{array}
\]

This is important: verse and verse* are treated as two different environments, the same holds for all starred verse like environments! If you want to recall the chords of a different type of environment, then you can use the option recall-chords:
11. Transposing

Provided a song has the property key and the key is given as one of the twelve “usual” keys, i.e., one of the keys given in table 3, the chords of a song can be transposed.

\textbf{transpose} = \langle (number) \rangle

Transposes the chords of a song by \langle number \rangle of semitones. \langle number \rangle can be a negative number, then the chords are transposed down.

\textbf{enharmonic} = \text{sharp|flat}

Suppose you transpose a song in the key of E down a semitone. LEADSHEETS will then transpose to the key of Eb. It always chooses the key whose signature has less accidentals. You can force a variant, though, by using this option. With \text{enharmonic} = \langle \text{sharp} \rangle \text{LEADSHEETS} would have chosen D\# instead of Eb.
11. Transposing

\texttt{transpose-capo = true|false}

When this is set to true chords are transposed down on semitone per capo fret.

The transposing mechanism relies on the chords input syntax which means that if you change \texttt{chord-cs} horrible things may happen. \textit{So don’t change \texttt{chord-cs} and use \texttt{transpose} at the same time!}

<table>
<thead>
<tr>
<th>Key Input</th>
<th>Key Input</th>
<th>Key Input</th>
<th>Key Input</th>
<th>Key Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>C</td>
<td>G</td>
<td>F</td>
<td>F</td>
</tr>
<tr>
<td>G</td>
<td>G</td>
<td>D</td>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td>A</td>
<td>A</td>
<td>E</td>
<td>Eb</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>E</td>
<td>Ab</td>
<td>Ab</td>
<td>C</td>
</tr>
<tr>
<td>B</td>
<td>B</td>
<td>Db</td>
<td>Db</td>
<td>C</td>
</tr>
<tr>
<td>F#</td>
<td>F#</td>
<td>Gb</td>
<td>Gb</td>
<td>D</td>
</tr>
</tbody>
</table>

\begin{song}{transpose=2}{title={Mary Had A Little Lamb}, music={Stevie Ray Vaughan}, lyrics={traditional}, tags={srv,blues},key=E}
\begin{verse}
\(^{E7}\) Mary had a little lamb \(^{A7}\) \\
Its fleece was white as \(^{E7}\)snow, yea \(^{A7}\) \\
Everywhere the child \(^{B7}\)went \(^{A7}\) \\
The little lamb was sure to \(^{E7}\)go, yea
\end{verse}
\begin{verse}
\(^{E7}\) He followed her to school one day \(^{A7}\) \\
And broke the teacher’s \(^{E7}\)rule \(^{A7}\) \\
What a time did they \(^{E7}\)have \(^{A7}\) \\
That day at \(^{E7}\)school
\end{verse}
\end{song}
12. Other verse-like Environments

Songs can have lots of different kinds of parts: verses, choruses, bridges, intros, outros, and so on. Typographically they’re all more or less the same, at least for the purpose of this package. This means we’d ideally have environments for all of these parts with a distinct name in order to get a clean source. At the same time these environments should all behave basically the same. This is what the environments described in the following sections are for.

12.1. Available Environments

\begin{chorus}[\langle options\rangle]
An environment for specifying the chorus of a song.
\end{chorus}

\begin{chorus*}[\langle options\rangle]
The same as \texttt{chorus} but does not display the label.
\end{chorus*}

\begin{intro}[\langle options\rangle]
An environment for specifying the intro of a song.
\end{intro}

\begin{intro*}[\langle options\rangle]
The same as \texttt{intro} but does not display the label.
\end{intro*}

\begin{interlude}[\langle options\rangle]
An environment for specifying an interlude of a song.
\end{interlude}

\begin{bridge}[\langle bridge\rangle]
An environment for specifying a bridge of a song.
\end{bridge}
\begin{info}{(options)}
An environment for specifying arbitrary information. This environment has no label.
\end{info}

\begin{solo}{(options)}
An environment for specifying a solo to a song.
\end{solo}

\begin{solo*}{(options)}
The same as solo but does not display the label.
\end{solo*}

\section{Own verse-like Environments}

All environments mentioned in the previous section were defined with this command:

\begin{newversetype*}{(environment name)}{(default options)}
This defines an environment \textit{(environment name)}. This environment uses the verse-type template \textit{itemize} (see section 14.2) unless specified differently in the \textit{(default options)}. The starred version defines an environment which hides the displayed name. More precisely: the macro \texttt{verselabel} that is used in verse-type template definitions prints nothing in an environment defined with \texttt{newversetype*}.

This also defines a translation string (see section 15) \texttt{leadsheets/(environment name)} with both an empty translation fallback and English translation unless specified differently with the \texttt{name} option.

At last this defines a new counter \textit{(environment name)} and sets \texttt{the/(environment name)} to \texttt{arabic/(environment name)}.

As mentioned before in section 9.1 all environments defined this way have the options \texttt{format} and \texttt{label-format}. They have more options. Here’s a complete list:

\texttt{format} = \{(code)\} \hspace{1cm} \text{(initially empty)}
\texttt{(code)} is inserted at the beginning of the environment and can thus be used to add formatting, e.g., \texttt{format} = \{\itshape\}.

\texttt{label-format} = \{(code)\} \hspace{1cm} \text{(initially empty)}
The same for the environment labels.

\texttt{after-label} = \{(code)\} \hspace{1cm} \text{Default: :}
\texttt{(code)} is inserted in the label after the label text.

\texttt{name} = \{(name)\} \hspace{1cm} \text{(initially empty)}
The translation fallback and English translation of the environment. This should only be used with \texttt{newversetype}. Later changes should be made with \texttt{DeclareTranslation} (see section 15).

\texttt{template} = \{(template)\} \hspace{1cm} \text{Default: \texttt{itemize}}
The verse-type template used for the environment, see section 14.2 for details.
12. Other verse-like Environments

**numbered** = true|false  
Default: false
If set to true \verselabel{} (used in verse-type template definitions, see section 14.2) adds a number after the name.

**named** = true|false  
Default: true
If set to true \verselabel{} (used in verse-type template definitions, see section 14.2) prints the name of the current environment (as determined by the translation of the corresponding string, see also section 15).

**recall-chords** = \{\langle environment\rangle-\langle class\rangle\}
An option to be used with the remember-chords mechanism, see section 10.4 for an example. If you’re not using different classes then \{class\} needs to be default, see also page 21. The \texttt{class} to a verse-like environment allows for example to have verses with different chords which are still counted regularly together with the remember-chords option.

There are also a number of general options for setting the defaults of some options for all environments:

**verses-format** = \{\langle code\rangle\}  
(initially empty)
Sets the format for all verse like environments.

**verses-label-format** = \{\langle code\rangle\}  
(initially empty)
Sets the label format for all verse like environments.

**verses-after-label** = \{\langle code\rangle\}  
Default: :
Default \{code\} that is inserted in the label after the label text of verse like environments.

This options only determine the formatting of an environment if the corresponding options of the environment hasn’t been set.

Let’s summarize: the label text of these environments is built of three items in the following order:

1. The \{code\} set with the corresponding label-format option.
2. The label text as defined as second argument to \newversetype{} or as declared through the corresponding translation.
3. The \{code\} set with the corresponding after-label option.

\begin{verbatim}
\newversetype{foo}{Foo}
\setleadsheets{
  foo/label-format = \bfseries ,
  foo/after-label = ~\Rightarrow$
}
\begin{foo}
Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\$
  sed diam nonumy eirmod tempor invidunt ut labore et dolore\$
\end{foo}
\end{verbatim}
And just to give you some more examples here is how some of the existing environments are defined:

\begin{verbatim}
newversetype{verse}[ name=Verse, named=false, after-label= ]
newversetype*{verse*}
newversetype{chorus}[ name=Chorus ]
newversetype*{chorus*}
\end{verbatim}

13. Typesetting Bars

Sometimes it can be useful to typeset the chord scheme of a song. Then one should be able to indicate start and beginnings of bars, maybe indicate repeats and so on. While this is obviously possible with the macros provided by the \texttt{musicsymbols} package listed in table 2 it may be more convenient to have a shorter syntax. This is why inside the \texttt{song} environment some characters can be made (or are) active (see section 8.5). For the typesetting of bars this are the characters \texttt{: nor |}. Per default they are not active, though. If you want to use the shortcut syntax you have to use the option \texttt{bar-shortcuts}. Here’s a short example that emulates the behaviour by setting the characters active explicitly:

\begin{verbatim}
catcode`|=\active
\catcode`=:\active
|: repeat | this |: and | this :| \par
| this | part | ends | here || \par
| the | song | is over | now |||
\end{verbatim}

\begin{verbatim}
| repeat |this || and |this ||
|this|part|ends|here ||
|the|song|is over|now ||
\end{verbatim}
14. Templates

All possibly combinations that have a special definition are shown in the example above. The replacements that are done internally are these:

| – \normalbar \space (the space is there because otherwise it eats following spaces which would be annoying)
|: – \leftrepeat
|:: – \rightrepeat
|::: – \leftrightrepeat
|| – \doublebar
||| – \stopbar

14. Templates

14.1. Title Templates

14.1.1. Background

The titles of songs set with the song environment are displayed according to the chosen title template. It is chosen through the option title-template which can be set with \setleadsheets or as option to a specific song environment. LEADSHEETS provides few predefined templates and an easy mechanism to define own templates.

14.1.2. Existing Templates

Currently LEADSHEETS provides two templates:

minimal  This only typesettes the song title in a \section*.

tabular  This typesets the song title in a \section and prints some song properties in a \tabular below it. This template needs the array [MC19] package loaded.

14.1.3. Own Templates

The principle is pretty straight forward: templates are defined with the following command:

\definesongtitletemplate{(name)}{(code)}

This defines the template \texttt{name}.

Inside of \texttt{code} any code can be used. The idea is that you use the commands presented below to insert song properties where you want them.

First there are two commands related to defining new properties:

\definesongproperty{(property)}

Defines a new property \texttt{property}. All existing properties have been defined this way. The command can only be used in the preamble.
14. Templates

\copyprop{\textit{from}}{\textit{to}}
Copies the values of property \textit{from} to property \textit{to} if property \textit{to} has not been set but property \textit{from} has been. For example all sort-\textit{property} properties have been treated this way so they have the \textit{property} value as fallback. The command can only be used in the preamble.

Then there are a number of commands related to retrieving and using the values of properties. All these commands only make sense inside a title template definition (see section 14.1). Some of the commands are expandable which means they can be used in an \edef like context, \textit{i.e.}, they are also suitable for writing the property values to the table of contents or other auxiliary files.

\* \songprop{\textit{property}}
Retrieves property \textit{property}.

\printsongprop{
\texttt{\textbackslash code}}{\textit{property}}{\textit{between two}}{\textit{between more}}{\textit{between last two}}
Default: \texttt{\@firstofone}
Prints a property list \textit{property} separated with \textit{between two} if the list contains only two items and separated with \textit{between more} and \textit{between last two} if the list contains more than two items. \texttt{\textbackslash code} is placed directly in front of each item and items are surrounded with braces which means that the last token in \texttt{\textbackslash code} may be a macro with a mandatory argument.

\usesongprop{
\texttt{\textbackslash code}}{\textit{property}}{\textit{between}}
Default: \texttt{\@firstofone}
Like \texttt{\printsongprop} but separates items with \textit{between} regardless of the length of the list.

\* \forsongprop{
\textit{property}}{\texttt{\textbackslash code}}
Places all items of the property list \textit{property} in the input stream, each item preceded with \texttt{\textbackslash code}. Items are surrounded with braces which means that the last token in \texttt{\textbackslash code} may be a macro with a mandatory argument.

\* \ifsongprop{
\textit{property}}{\texttt{\textbackslash true}}{\texttt{\textbackslash false}}
Checks if property \textit{property} has been set.

\* \ifanysongprop{
\texttt{csv list of properties}}{\texttt{\textbackslash true}}{\texttt{\textbackslash false}}
Checks if any of the properties in \texttt{csv list of properties} has been set.

\* \ifallsongprop{
\texttt{csv list of properties}}{\texttt{\textbackslash true}}{\texttt{\textbackslash false}}
Checks if all of the properties in \texttt{csv list of properties} have been set.

\ifsongpropsequal{
\textit{property 1}}{\textit{property 2}}{\texttt{\textbackslash true}}{\texttt{\textbackslash false}}
Checks if properties \textit{property 1} and \textit{property 2} have been set to the same value.

\ifsongmeasuring{\texttt{\textbackslash true}}{\texttt{\textbackslash false}}
\texttt{Leadsheets} measures the height of a song body before it typesets it and it can often be necessary in a template to know if the measuring phase is active or not. For example the song property \textit{height} should only be used if \texttt{not} in the measuring phase: its value gets determined there and is not yet available.
Exhaustively expands \texttt{\code}. Experienced users won’t need this. It is essentially `{\begingroup\edef\x{\endgroup\code}\x}\use:x`.

(More precisely it is a wrapper for the expl3 function \texttt{\use:x}.) This means that any \texttt{#} needs to be doubled. Inside the argument of this command non-robust macros that should not be expanded need to be prefixed with \texttt{\noexpand}.

With the right template definition you can index composers, interprets, song titles, ... You can write tables of contents for properties such as song titles, and so on. \texttt{LEADSHEETS} does not do this for you and it may require some experience to create templates which do all this.

### 14.1.4. Examples

In order to give you an idea on how to use songtitle templates I’ll show you how the existing ones are defined and one new definition.

**The minimal template**  This is quite short and self-explaining.

\begin{verbatim}
\definesongtitletemplate{minimal}{\section*{\songproperty{title}}}
\end{verbatim}

**A custom template**  Now let’s see an example for a newly defined template. It’s nearly as simple as the minimal template.

\begin{verbatim}
\documentclass{article}
\usepackage{leadsheets}
\definesongtitletemplate{custom}{
  \ifsongmeasuring
  {\section*}
  {\section}{%
  \songproperty{title}%
  \ifsongproperty{music}
  { (music by \printsongpropertylist{music}{ \& }{, }{ \& })%}
  %
  \}
  \setleadsheets{title-template = custom}
\begin{document}
\begin{song}{title={Mary Had A Little Lamb}, music={Stevie Ray Vaughan},
  lyrics={traditional}, tags={srv,blues}}
\end{song}
\end{document}
\end{verbatim}
Mary Had A Little Lamb (music by Stevie Ray Vaughan)

Mary had a little lamb
Its fleece was white as snow, yea
Everywhere the child went
The little lamb was sure to go, yea
He followed her to school one day
And broke the teacher’s rule
What a time did they have
That day at school

The *tabular* template This one is a lot more advanced and demonstrates various of the available commands.
A song using the tabular template:
1 Mary Had A Little Lamb

as interpreted by Stevie Ray Vaughan

Genre: blues
key: E

\[ E^7 \]
Mary had a little lamb
\[ A^7 \]
Its fleece was white as snow, yea
\[ B^7 \]
Everywhere the child went
\[ E^7 \]
The little lamb was sure to go, yea
\[ E^7 \]
He followed her to school one day
\[ A^7 \]
14. Templates

14.2. Verse-type Templates

14.2.1. Background

Similar to the songtitles also the verse like environments are typeset using templates. Defining them is just as easy as for the song titles.

14.2.2. Existing Templates

Currently LEADSHEETS provides only one template:

*itemize* Uses an itemize environment for typesetting the corresponding environment.

14.2.3. Own Templates

Own templates can be defined using these commands:

\defineversetypetemplate{(name)}{(begin code)}{(end code)}

This defines the template \textit{(name)}.

\verselabel

Used inside \defineversetypetemplate. This determines where the label of the environment using the template will be displayed.

\verselabelformat

Used inside \defineversetypetemplate. The format of the current environment as set with the corresponding \texttt{format} option.

\verseafterlabel

Used inside \defineversetypetemplate. The format of the current environment as set with the corresponding \texttt{after-label} option.

\versename

Used inside \defineversetypetemplate. This prints the name of the current environment.

\versenumber

Used inside \defineversetypetemplate. Expands to the \texttt{the(environment)} command for the current environment.

\ifversestarred{(true)}{(false)}

Can be used inside \defineversetypetemplate for checking if the current environment was defined by the starred version of \texttt{newversetype}.

\ifversenumbered{(true)}{(false)}

Can be used inside \defineversetypetemplate for checking if for the current environment the option \texttt{numbered} is true or false.

\ifversenamed{(true)}{(false)}

Can be used inside \defineversetypetemplate for checking if for the current environment the option \texttt{named} is true or false.
14. Templates

`\ifobeylines{⟨true⟩}{⟨false⟩}`
Checks if for the current song the option `obey-lines` is true or false.

Since with `\defineversetypetemplate` you define a template for an environment it has two argument for code: one for code at the beginning of the environment and one for code at the end. The command `\verselabel` internally uses the conditionals. Its definition is equivalent to the following:

\begin{verbatim}
\newcommand*{\verselabel}{%
  \ifversestarred
    {}\%
  {\verselabelformat
    \ifversenamed
      \versename
      \ifversenumbered{ }{}
    }
    {\ifversenumbered{ }{}
      \versenumber
    }\%
    \verseafterlabel
  }%
}\end{verbatim}

14.2.4. Examples

In order to give you an idea on how to use verse-type templates I’ll show you how the existing ones are defined and a few new definitions.

**The itemize template** This is how the *itemize* is defined.

\begin{verbatim}
\makeatletter
\defineversetypetemplate{itemize}{%
  \itemize
  @itemdepth=0
  \ifobeylines
    \setlength{\parskip}{0pt}\%
    \setleadsheets{ obey-lines-parskip = \parsep }%
  \else
    \setlength{\parskip}{\itemsep}\%
  \fi
}\end{verbatim}
The most interesting part is probably the \texttt{\ifobeylines} part. When the option \texttt{obey-lines} is set to true an end of a line inserts a \texttt{par} token. So in order not to get a vertical skip after every line the template sets \texttt{parskip} to zero. With \texttt{obey-lines = {true}} an empty line also inserts a \texttt{par} token but it also inserts a vertical space according to the value set with \texttt{obey-lines-parskip}. This option can only be set in a verse-type template definition (which is why it isn’t documented elsewhere). All verse like environments initialize the length to the current value of \texttt{parskip before} the template code is inserted.

A \texttt{flushleft} template  An example for a template \texttt{flushleft} that typesets the label in the margin:

\begin{verbatim}
\defineversetypetemplate{flushleft}
\{%
\noindent\llap{\verselabel\space}%
\flushleft
\unskip
\vspace*{-\baselineskip}
\ifobeylines
{%
\setlength\parskip{0pt}
\setleadsheets{ obey-lines-parskip = .5\baselineskip }%
%
\setlength\parskip{.5\baselineskip}
\vspace*{-\parskip}
%}
\}
\endflushleft
\begin{chorus}[template=flushleft]
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \\ 
sed diam nonumy eirmod tempor invidunt ut labore et dolore \\ 
magna aliquyam erat, sed diam voluptua.
\end{chorus}
\end{verbatim}
14. Templates

Chorus: Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

A \textit{flushright} template \quad An example for a template \textit{flushright} that typesets the label in the margin:

\begin{flushright}
\begin{verbatim}
defineversetypetemplate{flushright}
  \noindent\llap{\verselabel\space}\flushright\unskip\vspace*{-\baselineskip}
  \ifobeylines{\setlength\parskip{0pt}\setleadsheets{ obey-lines-parskip = .5\baselineskip }\}
  \setlength\parskip{.5\baselineskip}\vspace*{-\parskip}
\end{flushright}
\begin{chorus}[template=flushright]
  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \\sed diam nonumy eirmod tempor invidunt ut labore et dolore \\magna aliquyam erat, sed diam voluptua.
  \begin{flushright}
  Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \\sed diam nonumy eirmod tempor invidunt ut labore et dolore \\magna aliquyam erat, sed diam voluptua.
  \end{flushright}
\end{chorus}
\end{verbatim}
\end{flushright}

Chorus: 

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore
14. Templates

As you can see it’s not entirely easy to define a template that suits both songs with and without \texttt{obey-lines = \{true\}}. Personally I would forget about that option and not care about it at all in my templates.

**A framed template** Last but not least an example using the \texttt{mdframed} package [DS13] – just to show you that everything is possible. The example adapts one of the examples of \texttt{mdframed}’s manual.

\begin{verbatim}
defineversetypetemplate{framed}
\begin{verbatim}
\{
\begin{mdframed}[frametitle={
\tikz[baseline=(current bounding box.east),outer sep=0pt]
\node[anchor=east,rectangle,fill=blue!20,rounded corners=2pt]{\strut\verselabel};},
roundcorner = 5pt ,
linemargin = 1pt ,
linewidth = 2pt,
topline = true,
frametitleaboveskip = \dimexpr-\ht\strutbox\relax ,
]\end{mdframed}
\end{verbatim}
\end{verbatim}

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \textit{sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.}

\end{verbatim}

\begin{chorus}{template=framed}
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \textit{sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.}

Lorem ipsum dolor sit amet, consetetur sadipscing elitr, \textit{sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua.}

\end{chorus}
15. Internationalization

The environments described in sections 9 and 12 as well as a few other words used in \texttt{leadsheets} are translated with the help of the \texttt{translations} package \cite{Nie20}. All predefined and available translation strings are listed in table 4. You can change those translations or add translations for other languages with this command:

\begin{verbatim}
\DeclareTranslation{(language)}{(string)}{(translation)}
\end{verbatim}

The command provided by the \texttt{translations} package for translating strings.

Those translations can be used for example in song title templates (see section 14.1). One of the strings listed in table 4 is a little different: the string \texttt{leadsheets/interpret} is declared as

\begin{verbatim}
1. \DeclareTranslation{English}{leadsheets/interpret}
2. {as interpreted by \texttt{\printsongpropertylist{interpret}}{ \& }{, }{ \& }{}}
3. \DeclareTranslation{German}{leadsheets/interpret}
4. {wie von \texttt{\printsongpropertylist{interpret}}{ \& }{, }{ \& }{ } interpretiert}
\end{verbatim}

which means it uses the song property \texttt{interpret}. As a consequence it only really can be used inside a song environment. In other cases as for example in table 4 the property part expands to nothing (but the spaces around it are of course there). Also keep in mind that \texttt{\printsongpropertylist} is not expandable.
## Table 4: Predefined translation strings.

<table>
<thead>
<tr>
<th>String</th>
<th>English</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>leadsheets/major</td>
<td>major</td>
<td>Dur</td>
</tr>
<tr>
<td>leadsheets/minor</td>
<td>minor</td>
<td>Moll</td>
</tr>
<tr>
<td>leadsheets/chorus</td>
<td>Chorus</td>
<td>Refrain</td>
</tr>
<tr>
<td>leadsheets/verse</td>
<td>Verse</td>
<td>Strophe</td>
</tr>
<tr>
<td>leadsheets/composer</td>
<td>composer</td>
<td>Komponist</td>
</tr>
<tr>
<td>leadsheets/lyrics</td>
<td>lyrics</td>
<td>Text</td>
</tr>
<tr>
<td>leadsheets/key</td>
<td>key</td>
<td>Tonart</td>
</tr>
<tr>
<td>leadsheets/capo</td>
<td>capo</td>
<td>Kapo</td>
</tr>
<tr>
<td>leadsheets/fret</td>
<td>fret</td>
<td>Bund</td>
</tr>
<tr>
<td>leadsheets/interpret</td>
<td>as interpreted by</td>
<td>wie von interpretiert</td>
</tr>
<tr>
<td>leadsheets/intro</td>
<td>Intro</td>
<td>Intro</td>
</tr>
<tr>
<td>leadsheets/interlude</td>
<td>Interlude</td>
<td>Interlude</td>
</tr>
<tr>
<td>leadsheets/bridge</td>
<td>Bridge</td>
<td>Bridge</td>
</tr>
<tr>
<td>leadsheets/solo</td>
<td>Solo</td>
<td>Solo</td>
</tr>
</tbody>
</table>
Part V.
The external Library and the leadsheet Class

16. The Idea

For some users – like myself – it may be desirable to create leadsheets that can be reused in another document as a compilation of songs. The usual way would be to

- either solve this manually by copying the song environment from one document to the other. This is not ideal, of course.
- place the song in a file of its own and \texttt{input} it in the different documents. This is easy and safe but requires a total of three files for two documents. That’s not really a problem, though, so it is a valid way.

\texttt{leadsheets} now provides a third way. You write the single leadsheet using the leadsheet class that comes with this package. Let’s call this file \texttt{single.tex} for the following discussion:

\begin{verbatim}
\documentclass{leadsheet}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\begin{document}
\begin{song}{ ... }
...
\end{song}
\end{document}
\end{verbatim}

You can now include this document as is in another file – let’s call it \texttt{multiple.tex}:

\begin{verbatim}
\documentclass{leadsheet}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\begin{document}
\begin{song}{ ... }
...
\end{song}
\end{document}
\end{verbatim}
17. The leadsheet Class

The leadsheet class is a class file built upon the scrartcl class. It defines a song title template \texttt{leadsheet}, makes a few setup choices

\begin{verbatim}
\setleadsheets{
    title-template = leadsheet ,
    bar-shortcuts ,
    verse/numbered ,
    verses-format = ,
    verses-label-format = \bfseries ,
    info/format = \itshape ,
    chorus/format = \itshape
}
\end{verbatim}

and sets the header and the footer of the leadsheet pages.

\begin{verbatim}
\documentclass{leadsheet}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\usepackage[ngerman]{babel}
\begin{document}
\begin{song}{title={Mary Had A Little Lamb}, music={Stevie Ray Vaughan},
    interpret={Stevie Ray Vaughan}, lyrics={traditional}, tags={srv,blues}}
\end{song}
\end{document}
\end{verbatim}
Mary Had A Little Lamb

Text: traditional, Musik: Stevie Ray Vaughan, wie gespielt von: Stevie Ray Vaughan

1. Mary had a little lamb
   Its fleece was white as snow, yea
   Everywhere the child went
   The little lamb was sure to go, yea

2. He followed her to school one day
   And broke the teacher’s rule
   What a time did they have
   That day at school

The header contains “⟨song title⟩ ⟨interpret⟩” if the leadsheet has more than one page in total. Otherwise it is empty.

The footer contains “⟨page⟩ of ⟨pages⟩” if the leadsheet has more than one page in total. Otherwise it is empty.

The class provides the verse type prechorus:

\begin{prechorus}{⟨options⟩}
Environment for marking the pre-chorus of a song.

\instruction{⟨text⟩}
Short macro to add instructional text.
17. The leadsheet Class

\texttt{mkinstruction\{text\}}

The formatting macro used by \texttt{instruction}. You can redefine it in order to change the appearance. The default definition is:
\texttt{newcommand*mkinstruction\[1\]{\textit{#1}}}

\texttt{choir\{text\}}

Short macro to add (background) choir text.

\texttt{mkchoir\{text\}}

The formatting macro used by \texttt{choir}. You can redefine it in order to change the appearance. The default definition is:
\texttt{newcommand*mkchoir\[1\]{\texttt{instruction*leadsheettranslate\{choir\}: #1}}}

\texttt{lsenparen\{text\}}

Short macro to add parentheses around text. Used internally by \texttt{instruction}.

\texttt{mklsenparen\{openen parenthesis\}\{closing parenthesis\}\{text\}}

The formatting macro used by \texttt{lsenparen}. You can redefine it in order to change the parentheses. The default definition is:
\texttt{newcommand*mklsenparen\[3\]{\texttt{textup\{#1\}#3\textup\{#2\}}}

The definition needs to be expandable and to expand to exactly two tokens or braced groups, one for the opening parenthesis and the second for the closing parenthesis!

Here is an example that demonstrates \texttt{choir} and \texttt{instruction}. (The code of the example code has been saved in a file called single-leadsheet.tex. It is used later on again.)

\begin{verbatim}
\documentclass{leadsheet}
\usepackage[T1]{fontenc}
\usepackage[utf8]{inputenc}
\begin{document}
\begin{song}{title=foobar}
\begin{verse}
\texttt{instruction\{keyboard only\} \\\nLorem ipsum dolor sit amet, consetetur sadipscing \\\n\texttt{choir\{sed diam nonumy eirmod tempor invidunt ut labore et dolore\}}
\end{verse}
\begin{prechorus}
\texttt{instruction\{+ band\} \\\nLorem ipsum dolor sit amet, consetetur sadipscing}
\end{prechorus}
\end{song}
\end{document}
\end{verbatim}
18. The external Library

Loading the external library allows to include complete leadsheet documents (i.e., with preamble) into a main document. This requires the included leadsheet to use the leadsheet class (see section 17). When the file is included the whole preamble of said file will be discarded. This can be changed with an option but then `\usepackage` and `\RequirePackage` will still be disabled and simply do nothing in the included file.

```latex
\includeleadsheet*⟨{options}⟩⟨{file name}⟩
```

Include the leadsheet `⟨file name⟩`. `⟨options⟩` apply locally for the included file. The file `⟨file name⟩` may include a complete document preamble.

The next example shows the basic usage by inputting the example file mentioned on page 49. Notice that the formatting setup by the leadsheets is not used. Also the title template `leadsheet` is not defined in this process.
18. The external Library

Also notice that the prechorus environment and the \texttt{instruction} and \texttt{choir} macros \textit{are} available! \textit{All new macros defined in the leadsheet class and described in section 17 are defined.} If you do not want this then you can use the starred version of \texttt{includeleadsheet}. In this process the preamble of the included file is gobbled. It may be desirable not to have the preamble gobbled, though – because of custom macro definitions for example.

\texttt{gobble-preamble = true|false} \hspace{1cm} \textbf{Default: true}

If set to \texttt{false} \texttt{includeleadsheet} will not gobble the preamble of the file it inputs. Using this is dangerous: for it to work \texttt{usepackage} and \texttt{RequirePackage} have to be disabled while the file is included. So including the preamble may lead to more problems than it solves.

The included leadsheets do not have to be a complete document – a standard tex file only consisting of the \texttt{song} environments will work just as nicely. Files that are complete documents but are using another class than leadsheet will lead to errors, though.
Wrong is right.

Thelonious Monk

Part VI.
Appendix

A. References

url: https://www.ctan.org/pkg/mdframed/.

[Ham18] Kevin W. Hamlen. songs. version 3.1, Sept. 21, 2018 (or newer).
url: https://www.ctan.org/pkg/songs/.

url: https://www.ctan.org/pkg/l3kernel/.

url: https://www.ctan.org/pkg/l3packages/.

url: https://www.ctan.org/pkg/array/.

url: https://www.ctan.org/pkg/translations/.

[Rat10] Christopher Rath. songbook. version 4.5, Apr. 30, 2010 (or newer).
url: https://www.ctan.org/pkg/songbook/.


B. Index

Symbols
\after-title..................16 \bassclef.......................5
\align-chords..................23 \before-song....................16
\allabreve....................5 \break............................5
\altoclef.....................5 bridge (environment)........16, 29
\arr...........................18
\array (package)..............33
\aug..........................9
\capo........................18f.
\capo.........................18f.
\capo-nr......................19
\capo-nr-format...............19
C
\Carlisle, David.............33
INDEX

\choir ........................................ 49, 51
\chord .......................................... 19, 22 ff., 26
\chord-cs ....................................... 16, 24, 28
\chordname ..................................... 7, 24
\chordnames (library) .......................... 4
\chords (library) ................................ 3 f., 7, 11, 28
\chords (package option) ..................... 3
\chorus (environment) ......................... 15, 29
\chorus* (environment) ......................... 29
\class .......................................... 21, 31
\composer ...................................... 17
\copysongproperty ................................ 34
\counter ........................................ 18

D
\DeclareTranslation ................................ 30
\definesongproperty ................................ 33
\definesongtitletemplate .25 ff., 33, 35 f.
\defineversetypetemplate .39–43
\dim .............................................. 10
\disable-measuring ............................. 17 f.
\double-flat .................................... 9
\double-flat .................................... 9
\doublebar ...................................... 6, 33
\doubleflat ..................................... 5, 8 f.
\doublesharp .................................... 5, 8 f.

E
\eighthrest ...................................... 5
\empty-chord-dim ................................ 22 f.
\enharmonic ..................................... 7, 22, 27
\expandcode ..................................... 35, 37, 43
\external (library) ............................. 4, 50
\external (package option) .................... 4

F
\flat ............................................... 9
\flat .............................................. 5, 8 f.
\format ........................................... 9, 21, 30, 39
\forsongpropertylist .......................... 34
\full (package option) .......................... 3
\full-dim ........................................ 9

G
\genre ............................................ 18
\german-B ....................................... 10
\german-H ....................................... 10
\gobble-preamble ................................ 51

H
\half-dim ........................................ 9
\halfrest ........................................ 5

Hamlen, Kevin W. ................................ 2
\height ........................................... 17 f., 34
I
\ID .................................................. 18
\ifsongsongproperty ............................ 34
\ifanypropertysame ............................. 34
\ifobeylines .................................... 40–43
\ifsongmeasuring ................................ 34 ff.
\ifsongpropertiesequal ......................... 34
\ifexport ........................................ 34 f., 37
\ifversenamed ................................... 39 f.
\ifversenumbered ................................ 39 f.
\ifversestarred ................................... 39 f., 43
\includeleadsheet ............................... 47, 50 f.
\input (environment) ............................ 7, 10, 22
\instruction ...................................... 48 f., 51
\interbarwidth ................................... 6
\interlude (environment) ....................... 15, 29
\interpret ........................................ 18, 44
\intro (environment) ............................ 15, 29
\intro* (environment) ......................... 29
\itemize (verse-type template) ................ 30, 39 f.

K
\key ............................................... 18, 22, 27 f.

L
\lykernel (bundle) ............................... 2
\lpackages (bundle) ............................. 2
\label-format ..................................... 21, 30 f.
\leadsheet (songtitle template) ............... 47, 50 f.
\leadsheet (class) ............................... 46 f., 50 f.
\leadsheets (class) .............................. 4–50
\leadsheettranslate .............................. 49
\leftrepeat ....................................... 6, 33
\leftfrightrepeat ................................ 6, 33
\LPP ............................................... 2
\lsparen .......................................... 49
\lyrics ............................................ 17

M
\major ............................................ 10
\major-nine ...................................... 10
\major-seven ..................................... 10
\mdframed (package) ............................ 43
\meter ............................................. 6
\meterC ........................................... 5
\minimal (package option) ...................... 3
\minimal (songtitle template) ................. 15, 33, 35
\minor ............................................ 10
\Mittelbach, Frank ............................. 33

\mkchoir ........................................ 49
\mkinstruction .................................... 49
\mksenparen ..................................... 49
\mksenparen ..................................... 49
\musejazz (library) ............................. 5 f., 11
\musejazz (package option) .................... 4
\musicsymbol ..................................... 5
\musicsymbols (library) ....................... 3–6, 9, 32
\musicsymbols (package option) ............. 3
\musix ............................................ 5

N
\name ............................................. 30
\named ............................................ 31, 39
\natural .......................................... 5
\newversetypemode ................................ 30 ff., 39

Niederberger, Clemens ........................... 2, 44
\normalbar ....................................... 6, 33
\normalbarwidth ................................ 6
\numbered ........................................ 31, 39

O
\obeylines ...................................... 17, 24, 40 f., 43
\obey-lines ...................................... 41
\obeylines-parskip ................................ 41
\output-notation ................................ 7, 10, 22

P
\prechorus (environment) ...................... 48, 51
\print-chords .................................... 23
\print-tags ....................................... 16, 18
\printsongpropertylist ........................ 34 ff., 37, 44
\properties (library) ........................... 4

Q
\quarterrest ..................................... 5
\quarterrest ..................................... 5

R
\Rath, Christopher ............................. 2
\recall-chords ................................... 21, 26, 31
\remember-chords ............................... 21, 25, 31
\rightrepeat ..................................... 6, 33

S
\Schubert, Elke ................................. 43
\scartel (class) .................................. 47
\section .......................................... 15, 33
\setchords ....................................... 8–11, 37
\setleadsheets 9, 16, 20–23, 31, 33, 35, 38, 40–43, 47
\sharp .............................................. 9
\sharp .............................................. 5, 8 f.
\short-title ...................................... 17
\shorthands (library) .......................... 4
\sixteenth (library) ............................ 4

53
<table>
<thead>
<tr>
<th>Index Term</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>smash-chords</td>
<td>23</td>
</tr>
<tr>
<td>smash-next-chord</td>
<td>22 f.</td>
</tr>
<tr>
<td>solo (environment)</td>
<td>30</td>
</tr>
<tr>
<td>solo* (environment)</td>
<td>30</td>
</tr>
<tr>
<td>song (environment)</td>
<td>7, 13, 15–19, 22 f., 32 f., 44, 46, 51</td>
</tr>
<tr>
<td>song-form</td>
<td>16</td>
</tr>
<tr>
<td>songbook (package)</td>
<td>2</td>
</tr>
<tr>
<td>\songproperty</td>
<td>34–37</td>
</tr>
<tr>
<td>songs (library)</td>
<td>3 f.</td>
</tr>
<tr>
<td>songs (package option)</td>
<td>4</td>
</tr>
<tr>
<td>songs (package)</td>
<td>2</td>
</tr>
<tr>
<td>sort-arr</td>
<td>18</td>
</tr>
<tr>
<td>sort-band</td>
<td>18</td>
</tr>
<tr>
<td>sort-composer</td>
<td>17</td>
</tr>
<tr>
<td>sort-interpret</td>
<td>18</td>
</tr>
<tr>
<td>sort-lyrics</td>
<td>17</td>
</tr>
<tr>
<td>sort-short-title</td>
<td>17</td>
</tr>
<tr>
<td>\stopbar</td>
<td>6, 33</td>
</tr>
<tr>
<td>subtitle</td>
<td>17</td>
</tr>
<tr>
<td>sus</td>
<td>10</td>
</tr>
<tr>
<td>tabular (song template)</td>
<td>33, 37</td>
</tr>
<tr>
<td>tags</td>
<td>17 f.</td>
</tr>
<tr>
<td>templates (library)</td>
<td>4</td>
</tr>
<tr>
<td>tempo</td>
<td>18</td>
</tr>
<tr>
<td>text-format</td>
<td>16</td>
</tr>
<tr>
<td>\textmusix</td>
<td>5</td>
</tr>
<tr>
<td>The L\TeX3 Project Team</td>
<td>2</td>
</tr>
<tr>
<td>The Real Book</td>
<td>2</td>
</tr>
<tr>
<td>\thickbarwidth</td>
<td>6</td>
</tr>
<tr>
<td>title</td>
<td>17</td>
</tr>
<tr>
<td>title-template</td>
<td>16, 33</td>
</tr>
<tr>
<td>translations (library)</td>
<td>4</td>
</tr>
<tr>
<td>translations (package)</td>
<td>2, 44</td>
</tr>
<tr>
<td>transpose</td>
<td>7, 22, 27 f.</td>
</tr>
<tr>
<td>transpose-capo</td>
<td>28</td>
</tr>
<tr>
<td>transposing (library)</td>
<td>4</td>
</tr>
<tr>
<td>\trebleclef</td>
<td>4</td>
</tr>
<tr>
<td>\verselabel</td>
<td>30 f., 39–43</td>
</tr>
<tr>
<td>\versename</td>
<td>39 f.</td>
</tr>
<tr>
<td>\versenumber</td>
<td>39 f.</td>
</tr>
<tr>
<td>verses-after-label</td>
<td>39 f.</td>
</tr>
<tr>
<td>verses-form</td>
<td>21, 31</td>
</tr>
<tr>
<td>verses-label-format</td>
<td>21, 31</td>
</tr>
<tr>
<td>\wholerest</td>
<td>5</td>
</tr>
<tr>
<td>\useleadsheetslibraries</td>
<td>3, 5, 9, 47, 50</td>
</tr>
<tr>
<td>\usesongpropertylist</td>
<td>34</td>
</tr>
<tr>
<td>\verse (environment)</td>
<td>15, 17, 19, 21, 24 ff., 31, 39, 41</td>
</tr>
<tr>
<td>\verse* (environment)</td>
<td>19, 26</td>
</tr>
<tr>
<td>\verseafterlabel</td>
<td>39 f.</td>
</tr>
<tr>
<td>\verselabel</td>
<td>39 f.</td>
</tr>
<tr>
<td>\verselabelformat</td>
<td>39 f.</td>
</tr>
<tr>
<td>\versename</td>
<td>39 f.</td>
</tr>
<tr>
<td>\versenumber</td>
<td>39 f.</td>
</tr>
<tr>
<td>verses-after-label</td>
<td>31</td>
</tr>
<tr>
<td>verses-form</td>
<td>21, 31</td>
</tr>
<tr>
<td>verses-label-format</td>
<td>21, 31</td>
</tr>
<tr>
<td>\wholerest</td>
<td>5</td>
</tr>
<tr>
<td>\writechord</td>
<td>7 f., 10 ff., 19, 22 f., 37</td>
</tr>
<tr>
<td>writer</td>
<td>17</td>
</tr>
</tbody>
</table>