

LEADSHEETS

VO.0 2014/07/12

songbook (draft) (CN)

Clemens NIEDERBERGER

<https://github.com/cgnieder/leadsheets/>

contact@mychemistry.eu

Table of Contents

I. About the Bundle	2	6. The verse Environment	10
1. License and Requirements	2	6.1. Options	11
2. Background	2	6.2. Placing Chords	11
		6.3. Transposing	13
II. The MUSICSYMBOLS Package	2	7. Other verse-like Environments	15
		7.1. Available Environments	15
		7.2. Own verse-like Environments	16
III. The CHORDNAMES Package	3	8. Title Templates	16
3. The \chordname Command	3	8.1. Existing Templates	16
4. Options	4	8.2. Own Templates	16
		8.2.1. An Example	16
		8.2.2. Available Commands .	17
IV. The LEADSHEETS Package	5	9. Internationalization	18
5. The song Environment	6	V. Appendix	18
5.1. A First Example	6	A. References	18
5.2. Using the song Environment .	7	B. Index	19
5.3. Options	7		
5.4. Song Properties	8		

Part I.

About the Bundle

1. License and Requirements

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

LEADSHEETS requires the bundles `l3kernel` [The13a] and `l3packages` [The13b] to be available. It also needs the package translations [Nie13].

2. Background

Over the years I repeatedly wanted to typeset simple leadsheets of songs, *i. e.*, song lyrics and the corresponding chords.¹ This is not too hard with standard L^AT_EX 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` [Ham12] 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 bundle of packages is now the attempt to have a clean, documented and customisable version of my code.

Part II.




The **MUSICSYMBOLS** Package

The **MUSICSYMBOLS** package is a very small package that makes the music symbol font provided by MusiX_T_EX available as text font and then uses it to define a number of symbols. This redefines the macros `\sharp`, `\flat` and `\natural`. All defined symbols are listed in table 1.

MUSICSYMBOLS defines three further commands, namely `\musix`, `\textmusix` – a font switch and a text font command – and `\musicymbol`. Those commands are meant for internal use only which is why they’re not explained here.

1. I also have had the need to typeset leadsheets in the style of the *The Real Book* – but this is a task where other software than L^AT_EX is far easier.

TABLE 1: Symbols defined by **MUSICSYMBOLS**.

command	symbol	command	symbol
<code>\sharp</code>	#	<code>\flat</code>	b
<code>\doublesharp</code>	x	<code>\doubleflat</code>	bb
<code>\natural</code>	n		
<code>\trebleclef</code>		<code>\bassclef</code>	
<code>\altoclef</code>			
<code>\allabreve</code>	c	<code>\meterC</code>	¢
<code>\wholerest</code>	-	<code>\halfrest</code>	-
<code>\quarterrest</code>	z	<code>\eighthrest</code>	7
<code>\sixteenthrest</code>	z		

Part III.

The **CHORDNAMES** Package

3. The `\chordname` Command

CHORDNAMES provides the command `\chordname{⟨chord⟩}` for convenient typesetting of chords:

```

1 \chordname{Bb7(#9)} \chordname{Bbb9}      Bb7(#9) Bbb9 C#7(#9) Cx9
2 \chordname{C#7(#9)} \chordname{C##9}
```

`\chordname{⟨chord⟩}`

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

There are several token lists that are treated specially inside `\chordname`:

- ^ – This token is replaced by `\textsuperscript`.
- ma – The symbol for major chords. Per default this is empty. `\chordname{Gma}` G.
- mi – The symbol for minor chords. Per default this is m. `\chordname{Gmi}` Gm.
- o – The symbol for diminished chords. Per default this is `o`. `\chordname{Go}` G^o.

4. Options

- + – The symbol for augmented chords. Per default this is `+`. `\chordname{G+}` G^+ .
- /o – The symbol for half diminished chords. Per default this is `\textsuperscript{\o{}}`. `\chordname{G/o}` G^o .
- add – This is superscripted: `\chordname{Gadd9}` $G^{\text{add}9}$.
- sus – This is superscripted: `\chordname{Gsus4}` $G^{\text{sus}4}$.
- dim – This is superscripted: `\chordname{Gdim5}` $G^{\text{dim}5}$.
- maj7 – This is superscripted: `\chordname{Gmaj7}` $G^{\text{maj}7}$.
- maj9 – This is superscripted: `\chordname{Gmaj9}` $G^{\text{maj}9}$.

How these token lists are treated depends on optional settings:

```

1 \setchordnames{
2   major-seven = $\Delta$,
3   major-nine  = $\Delta\textsuperscript{9}$
4 }
5 \chordname{Gmaj7} \chordname{Gmaj9}
6 \chordname{G^6} \chordname{G6}
7 \chordname{G7^#5}

```

$G\Delta\ G\Delta^9\ G^6\ G^6\ G^{7\#5}$

4. Options

Options are set with the command

`\setchordnames{⟨options⟩}`

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

`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.

<code>double-flat</code> = { <i>code</i> }	Default: <code>\doubleflat</code>
The double flat symbol.	
<code>aug</code> = { <i>code</i> }	Default: <code>+</code>
The augmented symbol.	
<code>half-dim</code> = { <i>code</i> }	Default: <code>\o{}</code>
The half-diminished symbol.	
<code>full-dim</code> = { <i>code</i> }	Default: <code>o</code>
The diminished symbol.	
<code>dim</code> = { <i>code</i> }	Default: <code>dim</code>
The token list <code>dim</code> .	
<code>add</code> = { <i>code</i> }	Default: <code>add</code>
The token list <code>add</code> .	
<code>sus</code> = { <i>code</i> }	Default: <code>sus</code>
The token list <code>sus</code> .	
<code>major</code> = { <i>code</i> }	(initially empty)
The token list <code>ma</code> .	
<code>minor</code> = { <i>code</i> }	Default: <code>m</code>
The token list <code>mi</code> .	
<code>major-seven</code> = { <i>code</i> }	Default: <code>maj7</code>
The token list <code>maj7</code> .	
<code>major-nine</code> = { <i>code</i> }	Default: <code>maj9</code>
The token list <code>maj9</code> .	

Part IV.

The **LEADSHEETS** Package

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 5.4) that determine how the title of the song is typeset. For the typesetting of the titles a template mechanism is used (see section 8). Songs can also be tagged. The tags then allow to typeset only songs matching a list of tags that is specified via an option.

5. The song Environment

`\begin{song}[\langle options \rangle]{\langle properties \rangle}`

The main environment used to typeset songs.

5.1. A First Example

First let's take a look at an example:

```

1 \documentclass{article}
2 \usepackage{leadsheets}
3 \begin{document}
4
5 \begin{song}{title=Layla,composer=Eric Clapton,tags={clapton,unplugged,r&b}}
6 \begin{verse}
7   What will you do when you get lonely? \\
8   Noone waiting by your side. \\
9   You've been runnin', hidin' much too long. \\
10  You know it's just your foolish pride .
11 \end{verse}
12 \begin{chorus}
13   Layla, got me on my knees. \\
14   Layla, beggin' darlin', please! \\
15   Layla, darlin' won't you ease my worried mind?
16 \end{chorus}
17 \end{song}
18
19 \end{document}
```

Layla

What will you do when you get lonely?
 Noone waiting by your side.
 You've been runnin', hidin' much too long.
 You know it's just your foolish pride .

Chorus: Layla, got me on my knees.
 Layla, beggin' darlin', please!
 Layla, darlin' won't you ease my worried mind?

Per default the songtitle simply is a `\section*` without any other additions. This is the songtitle template “minimal”, see section 8 for more details on those templates and how to create your own.

5.2. 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 vers number and print it (as option to the internal `\item`).

`\begin{verse}[\langle options \rangle]`

An environment for specifying the verses of a song.

`\begin{chorus}[\langle options \rangle]`

An environment for specifying the chorus of a song.

This is the same as `\begin{verse}[type=chorus,\langle options \rangle]`.

`\begin{intro}[\langle options \rangle]`

An environment for specifying the intro of a song.

This is the same as `\begin{verse}[type=intro,\langle options \rangle]`.

`\begin{interlude}[\langle options \rangle]`

An environment for specifying an interlude of a song.

This is the same as `\begin{verse}[type=interlude,\langle options \rangle]`.

`\begin{bridge}[\langle bridge \rangle]`

An environment for specifying a bridge of a song.

This is the same as `\begin{verse}[type=bridge,\langle options \rangle]`.

These environments and their options are described in more detail in sections 6 and 7.

5.3. 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!*

`title-template = {\langle template name \rangle}`

Default: `minimal`

The songtitle template, see section 8 for details.

`chord-cs = {\langle cs \rangle}`

Default: `\chordname`

The command that is used to parse the chords. See section 6.2 for details. `\langle cs \rangle` needs to be a command that takes a mandatory argument.

5. The song Environment

`song-format` = { \langle TeX code \rangle } (initially empty)
 \langle TeX code \rangle is inserted *before* the song title at the beginning of the song environment.

`text-format` = { \langle TeX code \rangle } (initially empty)
 \langle TeX code \rangle is inserted *after* the song title at the beginning of the song environment.

`numbered` = `true` | `false` Default: `false`
Determines whether verses are numbered or not.

`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 .

5.4. Song Properties

Songs can have a number of properties which basically are used in songtitle templates (see section 8). One specific property, `tags`, plays a different role, though.

`title` = { \langle title \rangle }
This is the main title of the song.

`subtitle` = { \langle subtitle \rangle }
A subtitle.

`short-title` = { \langle short song title \rangle }
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` = { \langle song title \rangle }
If not set explicitly this property holds the same value as `title`.

`sort-short-title` = { \langle short song title \rangle }
If not set explicitly this property holds the same value as `short-title`.

`composer` = { \langle composer \rangle }
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` = { \langle composer \rangle }
If not set explicitly this property holds the same value as `composer`.

`lyrics` = { \langle writer \rangle }
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` = { \langle writer \rangle }
If not set explicitly this property holds the same value as `writer`.

5. The song Environment

`arr = {⟨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.

`sort-arr = {⟨arranger⟩}`

If not set explicitly this property holds the same value as `arr`.

`band = {⟨band⟩}`

The band who plays or played the song.

`sort-band = {⟨band⟩}`

If not set explicitly this property holds the same value as `band`.

`interpret = {⟨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.

`sort-interpret = {⟨interpret⟩}`

If not set explicitly this property holds the same value as `interpret`.

`genre = {⟨genre⟩}`

The genre of the song.

`key = {⟨key⟩}`

The key of the song. This property will be used when transposing will be implemented, see section 6.3.

`tempo = {⟨tempo⟩}`

The tempo of the song.

`tags = {⟨tags⟩}`

A comma separated list of tags. Those tags play a role for the option `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 two more properties, `counter` and `ID` that cannot be set but are set automatically for each song. The `counter` simply holds the number of the current song starting from 1 for the first song. The `ID` currently always is `song⟨counter⟩` where `⟨counter⟩` is the current `counter` value.

In principle all properties can get list of entries where items are separated with `_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 are written by Paul McCartney and John Lennon.²

It is possibly to define further such properties. For details see section 8.2.2.

2. This is not quite true: most songs were written either by Paul or John but legally usually both are the composers.

6. The verse Environment

`\begin{verse}[\langle options \rangle]`

An environment for specifying the verses of a song.

`\begin{verse*}[\langle options \rangle]`

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

```

1 \documentclass{article}
2 \usepackage{lead sheets}
3 \begin{document}
4
5 \begin{song}[numbered]{title=Foo}
6 \begin{verse}
7   Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\\
8   sed diam nonumy eirmod tempor invidunt ut labore et dolore\\
9   magna aliquyam erat, sed diam voluptua.
10 \end{verse}
11 \begin{verse*}
12   Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\\
13   sed diam nonumy eirmod tempor invidunt ut labore et dolore\\
14   magna aliquyam erat, sed diam voluptua.
15 \end{verse*}
16 \begin{verse}
17   Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\\
18   sed diam nonumy eirmod tempor invidunt ut labore et dolore\\
19   magna aliquyam erat, sed diam voluptua.
20 \end{verse}
21 \end{song}
22
23 \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.

6.1. Options

Currently the verse environment and all related environments only have one option:

`type = {⟨type⟩}`

The option determines what type of verse is to be typeset. For instance `type = {chorus}` will typeset a chorus. Which types are available will become clear in section 7.

```

1 \begin{verse}[type=chorus]
2   Lorem ipsum dolor sit amet, consetetur sadipscing elitr,\
3   sed diam nonumy eirmod tempor invidunt ut labore et dolore\
4   magna aliquyam erat, sed diam voluptua.
5 \end{verse}
```

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.

6.2. Placing Chords

Inside the song environment the character `^` is an active character which is defined to place a chord above words. More precisely is is equal to the following command:

`\chord* - {⟨chord⟩}⟨text⟩_`

Places `⟨chord⟩` centered above `⟨text⟩`. The starred version gobbles the trailing space while the unstarred version does not. Like the star the dash is optional. It sets the option `smash-next-chord` to `true`. `⟨text⟩` may be empty but the trailing space *must* be there. If `⟨text⟩` is empty then the chord is place above some horizontal space which can be set with the option `empty-chord-dim`.

6. The verse Environment

1	Text \chord{E7}text \chord*{B7}lon ger text	E ⁷ B ⁷ Text text longer text
---	--	--

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

1	text \chord{Gbmi7(b5)}text text	G m ^{7(b5)} text text text
---	---------------------------------	---

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

`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.

1	\setleadsheets{smash-next-chord=true}	G m ^{7(b5)} text text text
2	text \chord{Gbmi7(b5)}text text \par	G m ^{7(b5)}
3	text \chord{Gbmi7(b5)}text text \par	text text text
4	\setleadsheets{smash-chords=true}	G m ^{7(b5)}
5	text \chord{Gbmi7(b5)}text text \par	text text text
6	text \chord{Gbmi7(b5)}text text	G m ^{7(b5)} text text text

Before we forget:

`empty-chord-dim = {<dim>}`

Default: 1em

The horizontal space that is inserted if the `<text>` argument of `\chord` is empty.

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

```

1 \documentclass{article}
2 \usepackage{leadsheets}
3 \begin{document}
4
5 \begin{song}{title=Layla,composer=Eric Clapton,tags={clapton,unplugged,r&b}}
6 \begin{verse}
7   ^{C#mi7} What will you do when you get ^{G#7}lone ly? \\
8   ^{C#mi7} Noone ^{C}wai ting ^{D}by your ^{E}side. ^{E7} \\
9   ^{F#mi} You've been ^{B}run nin', ^{E}hid in' much too ^{A}long. \\
10  ^{F#mi} You know it's ^{B}just your foolish ^{E}pride .
11 \end{verse}
12 \begin{chorus}
13   ^{A}Lay ---^{Dmi7}la, \quad ^{Bb} ^{C}got me on my knees. \\

```

```

14 Lay^-{Dmi7}la, \quad ^{Bb} ^*{C}beg gin' darlin', ^{Dmi7}please, Layla. \\
15 Darlin' won't you ease my worried ^{Dmi7}mind? ^{Bb} ^{C}
16 \end{chorus}
17 \end{song}
18
19 \end{document}

```

Layla

$C\sharp m^7$ What will you do when you get lonely? $G\sharp^7$
 $C\sharp m^7$ C D E E^7
 Noone waiting by your side.
 $F\sharp m$ B E A
 You've been runnin', hidin' much too long.
 $F\sharp m$ B E
 You know it's just your foolish pride .

 A Dm^7 $B\flat$ C
 Chorus: Lay—la, got me on my knees.
 Dm^7 $B\flat$ C Dm^7
 Layla, beggin' darlin', please, Layla.
 Dm^7 $B\flat$ C
 Darlin' won't you ease my worried mind?

6.3. 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 2, the chords of a song can be transposed.

`transpose = {⟨number⟩}`

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.

`enharmonic = sharp|flat`

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

```

1 \documentclass{article}
2 \usepackage{leadsheets}
3 \begin{document}
4
5 \begin{song}[transpose=2]{
6   title=Layla,
7   composer=Eric Clapton,
8   tags={clapton,unplugged,r&b},

```

6. The verse Environment

```

9      key = Dmi
10   }
11   \begin{verse}
12     ^{C#mi7} What will you do when you get ^*{G#7}lone ly? \\
13     ^{C#mi7} Noone ^*{C}wai ting ^{D}by your ^{E}side. ^{E7} \\
14     ^{F#mi} You've been ^*{B}run nin', ^*{E}hid in' much too ^{A}long. \\
15     ^{F#mi} You know it's ^{B}just your foolish ^{E}pride .
16   \end{verse}
17   \begin{chorus}
18     ^*{A}Lay ---^{Dmi7}la, \quad ^{Bb} ^{C}got me on my knees. \\
19     Lay^{Dmi7}la, \quad ^{Bb} ^*{C}beg gin' darlin', ^{Dmi7}please, Layla. \\
20     Darlin' won't you ease my worried ^{Dmi7}mind? ^{Bb} ^{C}
21   \end{chorus}
22   \end{song}
23
24   \end{document}

```

Layla

D \sharp m⁷ A \sharp ⁷
What will you do when you get lonely?
D \sharp m⁷ D E F \sharp F \sharp ⁷
Noone waiting by your side.
G \sharp m B F \sharp B
You've been runnin', hidin' much too long.
G \sharp m B F \sharp
You know it's just your foolish pride .

B Em⁷ C D
Chorus: Lay—la, got me on my knees.
Em⁷ C D Em⁷
Layla, beggin' darlin', please, Layla.
Em⁷ C D
Darlin' won't you ease my worried mind?

TABLE 2: Allowed keys for the `key` property.

Key	Input	Key	Input	Key	Input	Key	Input
C	C			Am	A ^{mi}		
G	G	F	F	Em	E ^{mi}	Dm	D ^{mi}
D	D	B ^b	B ^b	Bm	B ^{mi}	Gm	G ^{mi}
A	A	E ^b	E ^b	F [♯] m	F [♯] ^{mi}	Cm	C ^{mi}
E	E	A ^b	A ^b	C [♯] m	C [♯] ^{mi}	Fm	F ^{mi}
B	B	D ^b	D ^b	G [♯] m	G [♯] ^{mi}	F [♯] m	F [♯] ^{mi}
F [♯]	F [♯]	G ^b	G ^b	D [♯] m	D [♯] ^{mi}	E [♯] m	E [♯] ^{mi}

7. 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.

7.1. Available Environments

`\begin{chorus}[\langle options \rangle]`

An environment for specifying the chorus of a song.

This is the same as `\begin{verse}[type=chorus,\langle options \rangle]`.

`\begin{chorus*}[\langle options \rangle]`

An environment for specifying the chorus of a song.

This is the same as `\begin{verse}[type=chorus*,\langle options \rangle]`. It does not display an introductory title to the left of the environment text.

`\begin{intro}[\langle options \rangle]`

An environment for specifying the intro of a song.

This is the same as `\begin{verse}[type=intro,\langle options \rangle]`.

`\begin{intro*}[\langle options \rangle]`

An environment for specifying the intro of a song.

This is the same as `\begin{verse}[type=intro*,\langle options \rangle]`. It does not display an introductory title to the left of the environment text.

`\begin{interlude}[\langle options \rangle]`

An environment for specifying an interlude of a song.

This is the same as `\begin{verse}[type=interlude,\langle options \rangle]`.

`\begin{bridge}[\langle bridge \rangle]`

An environment for specifying a bridge of a song.

This is the same as `\begin{verse}[type=bridge,\langle options \rangle]`.

`\begin{info}[\langle options \rangle]`

An environment for specifying arbitrary information.

This is the same as `\begin{verse}[type=info,\langle options \rangle]`. It does not display an introductory title to the left of the environment text.

`\begin{solo}[\langle options \rangle]`

An environment for specifying a solo to a song.

This is the same as `\begin{verse}[type=solo,\langle options \rangle]`.

`\begin{solo*}[\langle options \rangle]`

An environment for specifying a solo to a song.

This is the same as `\begin{verse}[type=solo*,<options>]`. It does not display an introductory title to the left of the environment text.

7.2. Own verse-like Environments

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

```
\newversetype*{<environment name>}{<displayed name>}
```

This defines an environment `<environment name>` with the text `<displayed name>` at the start of the environment left to the text. The starred version defines an environment which hides the displayed name. The environment also defines a translation string (see section 9) `leadsheets/<environment name>` with `<displayed name>` both as translation fallback and as English translation.

8. Title Templates

8.1. Existing Templates

8.2. Own Templates

8.2.1. An Example

```

1 \documentclass{article}
2 \usepackage{leadsheets}
3 \definesongtitletemplate{custom}{
4   \section{%
5     \songproperty{title}%
6     \ifsongproperty{composer}
7       { (by \songproperty{composer})}
8     }%
9   }
10 }
11 \setleadsheets{title-template = custom}
12 \begin{document}
13
14 \begin{song}{title=Layla,composer=Eric Clapton,tags={clapton,unplugged,r&b}}
15 \begin{verse}
16   What will you do when you get lonely? \\
17   Noone waiting by your side. \\
18   You've been runnin', hidin' much too long. \\
19   You know it's just your foolish pride .
20 \end{verse}
21 \begin{chorus}
22   Layla, got me on my knees. \\
23   Layla, beggin' darlin', please! \\
24   Layla, darlin' won't you ease my worried mind?
25 \end{chorus}
26 \end{song}

```



```

27
28 \end{document}

```

1 Layla (by Eric Clapton)

What will you do when you get lonely?
 Noone waiting by your side.
 You've been runnin', hidin' much too long.
 You know it's just your foolish pride .

Chorus: Layla, got me on my knees.
 Layla, beggin' darlin', please!
 Layla, darlin' won't you ease my worried mind?

8.2.2. Available Commands

First there are two commands related to defining new properties:

`\definesongproperty{⟨property⟩}`

Defines a new property `⟨property⟩`. All existing properties have been defined this way. The command can only be used in the preamble.

`\copysongproperty{⟨from⟩}{⟨to⟩}`

Copies the values of property `⟨from⟩` to property `⟨to⟩` if property `⟨to⟩` has not been set but property `⟨from⟩` has been. All sort-`⟨property⟩` properties have been treated this way. 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 template definition. Some of the commands are expandable which means they can be used in an `\edef` like context, *i. e.*, there are also suitable for writing the property values to the table of contents or other auxiliary files.

* `\songproperty{⟨property⟩}`

Retrieves property `⟨property⟩`.

`\printsongpropertylist[⟨code⟩]{⟨property⟩}{⟨between two⟩}{⟨between more⟩}{⟨between last two⟩}` Default: `\@firstofone`

Prints a property list `⟨property⟩` separated with `⟨between two⟩` if the list contains only two items and separated with `⟨between more⟩` and `⟨between last two⟩` if the list contains more than two items. `⟨code⟩` is placed directly in front of each item and items are surrounded with braces which means that the last token in `⟨code⟩` may be a macro with a mandatory argument.

`\usesongpropertylist`[$\langle code \rangle$]{ $\langle property \rangle$ }{ $\langle between \rangle$ } Default: `\@firstofone`
 Like `\printsongpropertylist` but separates items with $\langle between \rangle$ regardless of the length of the list.

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

* `\ifsongproperty`{ $\langle property \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }
 Checks if property $\langle property \rangle$ has been set.

`\ifsongpropertieequal`{ $\langle property 1 \rangle$ }{ $\langle property 2 \rangle$ }{ $\langle true \rangle$ }{ $\langle false \rangle$ }
 Checks if properties $\langle property 1 \rangle$ and $\langle property 2 \rangle$ have been set to the same value.

`\expandsongpropertycode`{ $\langle code \rangle$ }
 Exhaustively expands $\langle code \rangle$. Experienced user won't need this. It is essentially
`\begingroup\edef\x{\endgroup\langle code \rangle}\x.`

(More precisely it is a wrapper for the `expl3` function `\use:x`.) This means that any `#` needs to be doubled.

9. Internationalization

Part V.

Appendix

A. References

- [Ham12] Kevin W. HAMLEN. songs. version 2.14, Mar. 17, 2012.
 URL: <http://mirror.ctan.org/macros/latex/contrib/songs/>.
- [Nie13] Clemens NIEDERBERGER. translations. version 1.1a, Sept. 30, 2013.
 URL: <http://mirror.ctan.org/macros/latex/contrib/translations/>.
- [Rat10] Christopher RATH. songbook. version 4.5, Apr. 30, 2010.
 URL: <http://mirror.ctan.org/macros/latex/contrib/songbook/>.
- [The13a] THE L^AT_EX₃ PROJECT TEAM. l3kernel. version SVN 4582, July 28, 2013.
 URL: <http://mirror.ctan.org/macros/latex/contrib/l3kernel/>.
- [The13b] THE L^AT_EX₃ PROJECT TEAM. l3packages. version SVN 4582, July 28, 2013.
 URL: <http://mirror.ctan.org/macros/latex/contrib/l3packages/>.
- [Varoo] VARIOUS. *The Real Book. C Edition*. 6th edition. Vol. I.
 Hal Leonard Publishing Corporation, Jan. 1, 2000. ISBN: 978-0634060380.

B. Index

A	H	S
<code>add</code>5	<code>half-dim</code>5	<code>\section</code>6
<code>\allabreve</code>3	<code>\halfrest</code>3	<code>\setchordnames</code>4
<code>\altoclef</code>3	HAMLEN, Kevin.....2	<code>\setleadsheets</code>7, 12, 16
<code>aug</code>5	I	<code>sharp</code>4
B	<code>\ifsongpropertiesequal</code>18	<code>\sharp</code>2 ff.
<code>\bassclef</code>3	<code>\ifsongproperty</code>16, 18	<code>\sixteenthrest</code>3
<code>bridge</code> (environment).....7, 15	<code>info</code> (environment).....15	<code>smash-chords</code>12
C	<code>interlude</code> (environment).....7, 15	<code>smash-next-chord</code>11 f.
<code>\chord</code>11 f.	<code>intro</code> (environment).....7, 15	<code>solo</code> (environment).....15
<code>chord-cs</code>7	<code>intro*</code> (environment).....15	<code>solo*</code> (environment).....15
<code>\chordname</code>3 f.	L	<code>song</code> (environment).....5–8, 10–14, 16
<code>chorus</code> (environment).....6 f., 11–16	<code>l3kernel</code> (bundle).....2	<code>song-format</code>8
<code>chorus*</code> (environment).....15	<code>l3packages</code> (bundle).....2	<code>songbook</code> (package).....2
<code>\copysongproperty</code>17	LPPL.....2	<code>\songproperty</code>16 f.
D	M	<code>songs</code> (package).....2
<code>\definesongproperty</code>17	<code>major</code>5	<code>sus</code>5
<code>\definesongtitletemplate</code>16	<code>major-nine</code>5	T
<code>dim</code>5	<code>major-seven</code>5	<code>text-format</code>8
<code>double-flat</code>5	<code>\meterC</code>3	<code>\textmusix</code>2
<code>double-sharp</code>4	<code>minor</code>5	THE L ^A T _E X ₃ PROJECT TEAM.....2
<code>\doubleflat</code>3, 5	<code>\musicsymbol</code>2	<i>The Real Book</i>2
<code>\doublessharp</code>3 f.	<code>\musix</code>2	<code>title-template</code>7
E	N	<code>translations</code> (package).....2
<code>\eighthrest</code>3	<code>\natural</code>2 f.	<code>transpose</code>13
<code>empty-chord-dim</code>11 f.	<code>\newversetype</code>16	<code>\trebleclef</code>3
<code>enharmonic</code>13	NIEDERBERGER, Clemens.....2	<code>type</code>11
<code>\expandsongpropertycode</code>18	<code>numbered</code>8	U
F	P	<code>\usesongpropertylist</code>18
<code>flat</code>4	<code>print-tags</code>8 f.	V
<code>\flat</code>2 ff.	<code>\printsongpropertylist</code>17 f.	<code>verse</code> (environment).....6 f., 10 ff., 14, 16
<code>format</code>4	Q	<code>verse*</code> (environment).....10
<code>\forsongpropertylist</code>18	<code>\quarterrest</code>3	W
<code>full-dim</code>5	R	<code>\wholorest</code>3
	RATH, Christopher.....2	