A LaTeX PACKAGE

USER GUIDE

babelbidi

An extension to Babel's bidirectional support

LEFT TO RIGHT

RIGHT TO LEFT

Text direction · Bidirectional layout · Language selection · Font configuration

VERSION

1.3.0

November 27, 2026

MAINTAINER

Amer Amikhteh

LuaTeX (recommended) · XeTeX · pdfTeX

github.com/gambi-shah/babelbidi

Abstract

The babelbidi package extends Babel's bidirectional support by correcting specific direction-sensitive behaviour and providing interfaces for text direction, layout, language selection, and font configuration. These include directional boxes and controls for footnotes, lists, bibliographies, margin notes, and columns, as well as Latin-language citations and language selection for verbatim text. The package also supplies targeted compatibility patches for commonly used packages. It works with Babel under LaTeX (recommended), XeTeX, and PdfTeX, in documents with either an LTR or RTL main language.

Contents

1 Introduction 2

2 Interfaces and Engine Support 2
- 2.1 Engine Support . . . 2
- 2.2 Interface Overview . . . 3

3 Installing 3

4 Loading 3
- 4.1 Quick Start . . . 3
- 4.2 Package Load Order . . . 4
- 4.3 Engine Setup . . . 4
- 4.4 Package Options . . . 4
- 4.5 Subpackages . . . 5

5 Language Selection 6
- 5.1 Automatic Selection . . . 6
- 5.2 Explicit Selection . . . 6
- 5.3 Command and Environment Aliases 7

6 Text Direction 7
- 6.1 Inline Text . . . 7
- 6.2 Paragraphs and Blocks . . . 8

7 Boxes and Positioning 8
- 7.1 Directional Boxes . . . 8
- 7.2 Overlapping Text . . . 9

8 Footnotes and Title Notes 9
- 8.1 Footnote Direction . . . 9
- 8.2 Footnote Rules . . . 10
- 8.3 Extra Footnote Layouts . . . 11

9 Lists, Cases, and Counters 12
- 9.1 Lists and Bibliographies . . . 12
- 9.2 RTL Cases . . . 13
- 9.3 Counter Separators . . . 14

10 Margins and Columns 14
- 10.1 Margin Notes . . . 14
- 10.2 Two-Column Output . . . 14
- 10.3 Other Column Layouts . . . 15
- 10.4 Two-Column Table of Contents . . 15

11 Language and Font APIs 15
- 11.1 Font Selection . . . 15
- 11.2 Font Fallback . . . 16
- 11.3 Language Commands . . . 17
- 11.4 Language Environments . . . 17
- 11.5 Citations . . . 17

12 Engine Logos 18

13 Patches 18
- 13.1 Common Patches . . . 18
- 13.2 XeLaTeX Patches . . . 19
- 13.3 LaLaTeX Patches . . . 19

14 Minted Package 19
- 14.1 Unicode TeX Commands . . . 20
- 14.2 Language and Direction . . . 20
- 14.3 Character Directions . . . 20
- 14.4 Layout . . . 21

15 Limitations and Compatibility 21

16 License 21

17 Author and Maintainer 21

1 Introduction

Babel is a powerful, long-established framework for multilingual typesetting and localisation in LaTeX. It brings together language and script support, hyphenation, localised document conventions, language-specific fonts, and bidirectional typesetting, with support for pdfLaTeX, XeLaTeX, and LuaLaTeX. Its broad language coverage and extensible language model make it a strong foundation for multilingual documents.

Bidirectional documents can require additional control over direction-sensitive elements and their interaction with other packages. Footnotes, directional boxes, columns, and package-specific output are examples. babelbidi complements Babel in these areas, providing direction-control interfaces and targeted compatibility adjustments while retaining Babel's language and font configuration as the foundation. Babel's own APIs are used directly, not replaced by a parallel system.

One motivation for babelbidi is to provide a reusable Babel-based foundation for unipersian. More broadly, the aim is to adapt familiar typesetting interfaces, including those from XePersian, so that they can be used with different languages in a consistent Babel workflow.

The approach taken here is to build on Babel's language declarations and keep language selection distinct from direction control. This distinction allows a document to change direction without changing language, or to select a language together with its associated direction and fonts. Understanding which operation is needed is the starting point for using the interfaces in this guide.

This guide assumes basic familiarity with LaTeX and Babel's language declarations. Its organisation is intended to be familiar to readers of Babel's guide, without repeating the documentation of Babel's APIs. Begin with Sections 3–4.1 to install the package and prepare a document. The remaining sections serve as a reference for configuration and individual interfaces; engine-specific requirements and limitations are explained where they apply.

2 Interfaces and Engine Support

The package provides the complete user-level interfaces of bidi and luabidi, together with language and font interfaces built around Babel. Language-aware commands use the document's configured languages; direction-only commands leave the active language unchanged.

2.1 Engine Support

LuaLaTeX, XeLaTeX, and PdfLaTeX are supported, with either an LTR or RTL main language. LuaLaTeX is recommended for its broader capabilities and extensibility. XeTeX uses the bidi backend for direction and layout interfaces; LuaTeX and PdfTeX use implementations supplied by babelbidi.

The main engine-dependent capabilities are summarized below. Here, `Native` means that the feature is implemented by babelbidi, while `Via bidi` indicates that it is provided by bidi.

Feature	LuaTeX	XeTeX	PdfTeX
Direction and layout interfaces	Native	Via bidi	Native
Unicode font commands	Native	Native	–
Glyph fallback	Native	–	–
Extra footnote layouts	Native	Via bidi	Native
minted `textlang` and `textdir`	Native	Native	Native
minted `chardir`	Native	–	–

The table describes interface availability, not identical implementations, output, or language coverage. Unicode font commands require suitable OpenType fonts and fontspec. With PdfLaTeX, language support depends on suitable input and font encodings, fonts, and Babel language support. See also Section 15.

2.2 Interface Overview

The following table locates the main interface groups. Availability and behaviour depend on the engine as described in the referenced sections.

Interface group	Sections
Text direction and directional boxes	6, 7
Footnotes, title notes, and footnote rules	8
Citations, lists, bibliographies, cases, and counters	11.5, 9
Margin notes, columns, and table of contents	10
Language switching, fonts, and Latin verbatim text	11
Glyph fallback (LaTeX)	11.2
Engine logos	12
Package patches and minted options	13, 14

3 Installing

babelbidi is distributed through CTAN:

https://ctan.org/pkg/babelbidi

With TeX Live, install the package from a terminal or command prompt:

```
tlmgr install babelbidi
```

4 Loading

Load the package with:

```
\usepackage[<options>]{babelbidi}
```

Options not recognized by babelbidi are forwarded to Babel. In particular, `bidi` and `layout` are Babel options with engine-dependent defaults supplied by this package.

4.1 Quick Start

The following example works with LaTeX and XeLaTeX and requires the fonts Latin Modern Roman and Amiri:

```
\documentclass{article}
\usepackage{hyperref}
\usepackage{babelbidi}

\babelprovide[import,main]{english}
\babelprovide[import]{arabic}

\babelfont{rm}{Latin Modern Roman}
\babelfont[arabic]{rm}{Amiri}

\begin{document}
    English text with \rl{نص عربي}.

    \begin{otherlanguage}{arabic}
        نص عربي مع \lr{English text}.
    \end{otherlanguage}
\end{document}
```

The lowercase commands `\lr` and `\rl` select the configured LTR or RTL language, together with its direction and fonts. The uppercase commands `\LR` and `\RL` change direction only. Use the lowercase forms when the passage belongs to another language; use the uppercase forms when only its direction should change.

With LaTeX, Babel's `onchar` option can automate language and font selection from the characters in the text. To enable this in the example, replace the Arabic declaration with:

```
\babelprovide[import,mapdigits,onchar=ids fonts letters]{arabic}
```

Punctuation shared by several scripts, such as parentheses, may still require explicit language selection. Automatic selection does not remove the need to declare languages and assign suitable fonts.

4.2 Package Load Order

With XeLaTeX, babelbidi uses bidi by default. Both loading orders of hyperref and babelbidi are supported. Follow any load-order requirements documented by other packages.

4.3 Engine Setup

Normally, babelbidi loads Babel and chooses a bidirectional backend for the current engine. With LaTeX and beamer-rl, the class handles Babel setup instead.

Babel's `bidi` option selects the bidirectional backend. If it is not specified, babelbidi selects a default according to the engine:

Engine	Default value of `bidi`
LaTeX	`basic`
XeTeX	`bidi-r`
PdfTeX	`default`

With LaTeX, the default value of Babel's `layout` option is:

```
counters graphics footnotes lists
```

Specifying `layout` replaces this default; it does not add to it. For example:

```
layout={counters footnotes}
```

This selects only `counters` and `footnotes`. To retain `graphics` and `lists`, include them explicitly. Values containing multiple selectors must be enclosed in braces. The package applies this `layout` setting only with LaTeX; it has no effect in the XeTeX and PdfTeX loading branches.

For slides with an RTL main language, beamer-rl is recommended over beamer, although it is not required.

4.4 Package Options

The following keys assign languages to the four roles used by BabelBidi. Their automatic defaults and selection rules are described in Section 5.

`latin=`⟨*language*⟩

Assigns the Latin-script language, not necessarily English.

`ltr=`⟨*language*⟩

Assigns the LTR language, which need not use the Latin script.

`rtl=`⟨*language*⟩

Assigns the RTL language.

`code=`⟨*language*⟩

Assigns the language context for code and verbatim text, not a programming language or syntax-highlighting lexer.

`fontcache=`⟨*true*⟩/⟨*false*⟩

With LaTeX, controls caching of Babel font mappings used by automatic font selection and glyph fallback. It is enabled by default to improve compilation speed. In special cases, such as font changes made through the `rmfamily` hook, cached mappings may prevent changes from taking effect. Disable the cache when needed:

```
\usepackage[fontcache=false]{babelbidi}
```

`mintedunicode=`⟨*true*⟩/⟨*false*⟩

Controls the adjustment of Unicode command-name highlighting in minted `latex` code blocks and file listings with XeLaTeX and LuaLaTeX. The default is `true`; the adjustment is used only when minted is loaded. Set the option to `false` to disable it. See Section 14.1 for details and limitations.

`autofootnoterule`
`textwidthfootnoterule`

Select automatic footnote-rule placement or a full-width rule at package loading. The corresponding switches are described in Section 8.

`arabmaths`

With LaTeX, sets the mathematics direction to RTL. The option issues a warning with XeTeX and PdfTeX, where this LuaTeX direction setting is unavailable.

`extrafootnotefeatures=`⟨*on*⟩/⟨*off*⟩

Enables paragraph and multi-column footnote layouts. The default is `off`; specifying the option without a value enables it:

```
\usepackage[extrafootnotefeatures]{babelbidi}
```

The option also accepts explicit `=on` and `=off` values. With XeTeX, it is passed to bidi; with LaTeX and PdfTeX, babelbidi provides its own implementation. See Section 8.3 for the available commands.

`logostyle=`⟨*auto*⟩/⟨*simple*⟩

Selects the engine-logo style. The default is `auto`. Use `simple` for non-reflected logos, even when graphicx is loaded. See Section 12.

4.5 Subpackages

babelbidi loads babelbidi-lang and babelbidi-logos automatically. Both can also be used independently of the main package.

babelbidi-lang Provides the four language roles, `\lr`, `\rl`, `\latintoday`, the `latin` environments, and the command and environment aliases described in Section 5.3. For standalone use, load it after Babel; it accepts the same `latin`, `ltr`, `rtl`, and `code` keys listed above:

```
\usepackage[english]{babel}
\usepackage{babelbidi-lang}
```

Configure Babel's direction support and fonts as needed. This subpackage does not provide the main package's direction-only, font, layout, or compatibility interfaces.

babelbidi-logos Provides the engine logos documented in Section 12, without requiring Babel or babelbidi. When loaded directly, its option is named `style`, rather than `logostyle`; the values are `auto` (default) and `simple`:

```
\usepackage[style=simple]{babelbidi-logos}
```

amikhteh-logo An optional companion for the package-family mark used in this guide, not the engine logos. It is not loaded by babelbidi. Load it explicitly with `\usepackage{amikhteh-logo}`. Its `manual` option enables the optional page style; activate it with `\pagestyle{amikhteh-logo}`.

5 Language Selection

BabelBidi assigns Babel languages to four independent roles: `latin`, `ltr`, `rtl`, and `code`. A role assignment neither loads a language nor changes the active or main language; several roles may share one language. "The selected Latin language" and similar expressions refer to these roles, not the currently active language. The keys are documented in Section 4.

5.1 Automatic Selection

Unset roles are selected at package loading and after each `\babelprovide` call. Using Babel's script and direction metadata, the first applicable choice below is used:

`latin`: Latin-script main language; first loaded Latin-script language; `english`.

`ltr`: LTR main language; first loaded LTR language; `english`.

`rtl`: RTL main language; first loaded RTL language; `arabic`.

`code`: Latin-script main language; loaded `english`; first loaded Latin-script language; LTR main language; first loaded LTR language; main language (or `english` if it is Babel's `nil` placeholder).

5.2 Explicit Selection

Explicit keys override automatic choices independently and remain fixed when more languages are declared:

```
\usepackage[
latin=french, ltr=english, rtl=arabic, code=english
]{babelbidi}
\babelprovide[import,main]{arabic}
\babelprovide[import]{french}
\babelprovide[import]{english}
```

Declare every selected language through Babel before use, including automatic fallbacks. Establish declarations and role assignments before configuring fonts. For standalone use of babelbidi-lang, see Section 4.5.

5.3 Command and Environment Aliases

These commands accept names without a leading backslash and can be used to provide command names in another language.

`\eqcommand{⟨alias⟩}{⟨original⟩}`

Defines an alias that calls the original command, including any arguments. Later changes to the original command are reflected in the alias. For a TeX primitive, the alias is a direct copy.

`\eqenvironment{⟨alias⟩}{⟨original⟩}`

Defines an environment that uses the original environment. A matching starred alias is defined if the original starred form exists when the alias is declared or becomes available before the document begins.

`\EqEnvironment{⟨alias⟩}{⟨original⟩}`

Copies the original environment's begin and end commands as they are at the time of the declaration.

`\makezwnjletter`

Makes the zero-width non-joiner (U+200C) a letter for subsequent command names. This changes its category code in the current TeX group and requires XeLaTeX or LuaLaTeX.

```
\eqcommand{important}{textbf}
\eqenvironment{quotationexample}{quote}
\important{Text}
\begin{quotationexample}A quotation.\end{quotationexample}
```

6 Text Direction

These interfaces change direction while leaving the active language unchanged. They do not select another language's fonts or conventions. For text in another language, use the interfaces in Section 11.3 or Babel's own language commands. Use inline forms within a paragraph and switches or environments for whole paragraphs.

6.1 Inline Text

`\beginL`

Begins an explicit LTR text segment; it must be matched by `\endL`.

`\endL`

Ends the explicit LTR text segment.

`\beginR`

Begins an explicit RTL text segment; it must be matched by `\endR`.

`\endR`

Ends the explicit RTL text segment.

`\LR{⟨text⟩}` = `\LRE{⟨text⟩}`

Typesets the argument as inline LTR text. The two names are equivalent.

`\RL{⟨text⟩}` = `\RLE{⟨text⟩}`

Typesets the argument as inline RTL text. The two names are equivalent.

Prefer the argument-taking commands for short passages. With LaTeX and the default `basic` mode, `\LR` and `\RL` use explicit directions as in luabidi; Babel's automatic bidirectional processing is not applied within those passages.

6.2 Paragraphs and Blocks

`\setLTR` = `\setLR` = `\unsetRTL` = `\unsetRL`

These equivalent commands set paragraph direction to LTR.

`\setRTL` = `\setRL` = `\unsetLTR` = `\unsetLR`

These equivalent commands set paragraph direction to RTL.

The switches take no arguments; place them between paragraphs and use a group to limit their scope. End the final paragraph before closing the group. With LaTeX and the default `basic` mode, these switches and the directional block environments use explicit LuaTeX directions, following luabidi. Language selection remains available when Babel's automatic bidirectional processing is needed.

`\setRTLmain`

With LaTeX, sets the main page and paragraph direction to RTL. With XeTeX and PdfTeX, selects RTL as the main and current paragraph direction. It may be used in the preamble to start the document in RTL direction.

`\begin{LTR}` ... `\end{LTR}`
`\begin{RTL}` ... `\end{RTL}`

The `LTR` environment typesets an enclosed block in LTR direction; `RTL` typesets it in RTL direction. The environments handle paragraph boundaries and may contain several paragraphs.

`\begin{LTR*}` ... `\end{LTR*}`
`\begin{RTL*}` ... `\end{RTL*}`

The starred environments also set the block direction. With XeTeX, their paragraph indentation follows bidi; with LaTeX and PdfTeX, they locally set `\parindent` to zero.

7 Boxes and Positioning

7.1 Directional Boxes

These are directional variants of `\hbox`, `\vbox`, and `\vtop`; the suffix selects LTR or RTL direction.

`\hboxL`

Creates a horizontal LTR box without line breaking.

`\hboxR`

Creates a horizontal RTL box without line breaking.

`\vboxL`

Creates a vertical LTR box with a bottom reference point.

`\vboxR`

Creates a vertical RTL box with a bottom reference point.

`\vtopL`

Creates a vertical LTR box with a top reference point.

`\vtopR`

Creates a vertical RTL box with a top reference point.

Text in vertical boxes is broken into lines according to `\hsize`. For boxes containing lines of text, `\vbox` uses the last line's baseline and `\vtop` the first.

Before the opening brace, `to <dimension>` sets the final width of a horizontal box or height of a vertical box; `spread <dimension>` adds to its natural size. Stretchable or shrinkable space adjusts to the requested size. Use `\setbox` to store a box and `\box` to insert it.

7.2 Overlapping Text

These direction-sensitive variants of `\llap` and `\rlap` place text in a zero-width box.

`\bidillap{`⟨*text*⟩`}`

Overlaps text to the left in LTR text and to the right in RTL text.

`\bidirlap{`⟨*text*⟩`}`

Overlaps text to the right in LTR text and to the left in RTL text.

8 Footnotes and Title Notes

Footnote direction, rule appearance, and layout are separate settings. Choose an explicit command to direct one note, or a switch to set the direction of subsequent ordinary footnotes. Neither choice by itself changes the note's language.

8.1 Footnote Direction

The LTR and RTL forms select the direction of the footnote text, not its language. Use Babel's language commands when a footnote also requires a different language or font.

`\LTRfootnote[`⟨*num*⟩`]{`⟨*text*⟩`}`

Inserts a footnote whose text is set in LTR direction.

`\RTLfootnote[`⟨*num*⟩`]{`⟨*text*⟩`}`

Inserts a footnote whose text is set in RTL direction.

`\LTRfootnotetext[`⟨*num*⟩`]{`⟨*text*⟩`}`

Inserts LTR footnote text without a mark in the main text.

`\RTLfootnotetext[`⟨*num*⟩`]{`⟨*text*⟩`}`

Inserts RTL footnote text without a mark in the main text.

The optional ⟨*num*⟩ has the same meaning as in `\footnote` or `\footnotetext`. The `\LTRfootnotetext` and `\RTLfootnotetext` commands do not insert a mark in the main text; use them with `\footnotemark` when a mark is needed.

In LaTeX, these two directed footnote commands follow luabidi's number formatting. A `\label` inside a note without an optional number can therefore resolve to the current footnote number when referenced, rather than the original note number.

`\LTRthanks{`⟨*text*⟩`}`

Inserts a title note in LTR direction.

`\RTLthanks{`⟨*text*⟩`}`

Inserts a title note in RTL direction.

The title-note commands are counterparts of `\thanks` and can be used in the `\author` or `\title` argument.

`\setfootnoteLR` = `\unsetfootnoteRL`

These equivalent commands set subsequent ordinary footnotes to LTR.

`\setfootnoteRL` = `\unsetfootnoteLR`

These equivalent commands set subsequent ordinary footnotes to RTL.

The switches insert no note and take no arguments. Their direction setting is local to the current group and remains active until changed. The `unset` forms select the opposite direction; they do not restore Babel's initial behaviour.

Explicitly directed footnotes take precedence over the policy: for example, `\RTLfootnote` still produces an RTL footnote after `\setfootnoteLR`. Before a direction policy is selected, the existing Babel behaviour is retained.

8.2 Footnote Rules

The controls below position and size the rule above the footnote area. They do not change the direction of the footnote text.

`\leftfootnoterule` = `\LRfootnoterule`

These equivalent commands select a left-aligned footnote rule.

`\rightfootnoterule` = `\RLfootnoterule`

These equivalent commands select a right-aligned footnote rule.

`\textwidthfootnoterule`

Selects a full-width footnote rule.

`\autofootnoterule`

Selects rule alignment according to footnote direction. Automatic alignment is engine-dependent:

LaTeX: Inspect the direction of the first line in the ordinary footnote insertion box; use a left rule if no line is found.

pdfTeX: Use the first explicitly directed footnote recorded since the preceding shipout; use a left rule if none was recorded. This is a fallback, not an inspection of the final footnote box.

XeTeX: Use the implementation supplied by bidi.

`\footnoterulewidth`

Length controlling the width of short footnote rules. With LaTeX and pdfTeX, the initial short-rule width is `.4\columnwidth`, evaluated when the length is set.

For example:

```
\setlength{\footnoterulewidth}{.3\columnwidth}
\rightfootnoterule
```

`\SplitFootnoteRule`

Selects a full-width rule above a continued footnote. Select the ordinary rule before `\SplitFootnoteRule`. It retains that rule on ordinary pages and uses a full-width rule above a continuation from the preceding page.

`\debugfootnotedirection`

Enables direction messages when the automatic rule is evaluated. To diagnose automatic rule selection, use:

```
\autofootnoterule
\debugfootnotedirection
```

The LaTeX and pdfTeX implementations provide no corresponding public command to disable these diagnostic messages. Remove the diagnostic command for normal runs.

With LaTeX and pdfTeX, rule selectors make global assignments. In RTL-main documents, the package selects a right-aligned default rule only if no replacement rule has already been selected. Extra footnote layouts change rule handling as described below.

8.3 Extra Footnote Layouts

The `extrafootnotefeatures` package option enables the layouts below. With LaTeX and pdfTeX, they support page and minipage footnotes while preserving each note's language, direction, labels, and hyperlinks. The grouping and output-order restrictions described after the command list still apply.

`\normalfootnotes[*]`

Selects the ordinary one-column layout.

`\paragraphfootnotes[*]`

Selects paragraph layout.

`\twocolumnfootnotes[*]`

Selects a two-column layout.

`\threecolumnfootnotes[*]`

Selects a three-column layout.

`\fourcolumnfootnotes[*]`

Selects a four-column layout.

`\fivecolumnfootnotes[*]`

Selects a five-column layout.

`\sixcolumnfootnotes[*]`

Selects a six-column layout.

`\sevencolumnfootnotes[*]`

Selects a seven-column layout.

`\eightcolumnfootnotes[*]`

Selects an eight-column layout.

`\ninecolumnfootnotes[*]`

Selects a nine-column layout.

`\tencolumnfootnotes[*]`

Selects a ten-column layout.

Each layout command accepts an optional star. Without it, the footnote mark occupies a fixed-width box; with it, the box uses the mark's natural width:

```
\twocolumnfootnotes*
```

With LaTeX and pdfTeX, paragraph and multi-column layouts collect RTL and LTR notes separately. Output begins with ordinary one-column footnotes, followed by the RTL group and then the LTR group. Within a multi-column group, columns run right to left for RTL notes and left to right for LTR notes. The visual order may therefore differ from the order of note references in the text. With XeTeX, the layouts and their detailed behaviour are supplied by bidi.

With LaTeX and pdfTeX, a layout assignment is local, but the collected notes are assembled later. Select the layout in the preamble or at a page boundary, and keep it in force until the affected notes have been output. Do not combine separate paragraph and multi-column settings within the same pending stream. For a document-wide layout:

```
\usepackage[extrafootnotefeatures]{babelbidi}
\twocolumnfootnotes
```

With LaTeX and pdfTeX, to change the layout in the body, flush the preceding page first:

```
\clearpage
\paragraphfootnotes
```

With XeTeX, select `\paragraphfootnotes` in the preamble; its implementation in bidi requires preamble-only setup.

With LaTeX and pdfTeX, loading `extrafootnotefeatures` installs a rule spanning `\columnwidth` and disables `\autofootnoterule`, `\leftfootnoterule`, `\LRfootnoterule`, `\rightfootnoterule`, `\RLfootnoterule`, and `\textwidthfootnoterule`. Calling `\normalfootnotes` does not restore these selectors. The rule spans the footnote area of the current document column, not an individual footnote column.

9 Lists, Cases, and Counters

9.1 Lists and Bibliographies

The `LTRitems`, `RTLitems`, `LTRbibitems`, and `RTLbibitems` environments direct entries within an existing list or bibliography without changing the active language. They do not create the surrounding list or bibliography.

`\begin{LTRitems}` ... `\end{LTRitems}`

`\begin{RTLitems}` ... `\end{RTLitems}`

The `LTRitems` environment sets the direction of the enclosed list items to LTR; `RTLitems` sets it to RTL.

The environments group consecutive `\item` entries. They can be used inside lists such as `itemize` and `enumerate`; they do not replace those environments.

For example, a list can contain an RTL group between ordinary items:

```
\begin{itemize}
    \item The first item
    \begin{RTLitems}
        \item مورد دوم
        \item مورد سوم
    \end{RTLitems}
    \item The fourth item
\end{itemize}
```

`\begin{latinitems}` ... `\end{latinitems}`

Like `LTRitems`, this groups items within an existing list. It also selects the configured Latin-script language and its fonts, as the `latin` environment does. The previous language and direction are restored at the end of the group.

`\begin{LTRbibitems} ... \end{LTRbibitems}`

`\begin{RTLbibitems} ... \end{RTLbibitems}`

The `LTRbibitems` environment sets the direction of the enclosed bibliography entries to LTR; `RTLbibitems` sets it to RTL.

Place these environments inside `thebibliography`, using `\bibitem`, not `\item`, for each entry. The following example directs one entry without changing the bibliography heading:

```
\begin{thebibliography}{9}
    \begin{LTRbibitems}
        \bibitem{knuth}
        Donald E. Knuth. \textit{The TeXbook}. 1984.
    \end{LTRbibitems}
\end{thebibliography}
```

Here, 9 reserves space for a single-digit label, and `knuth` is the key used by `\cite{<knuth>}`.

To set the direction of all bibliography entries, use one of the following switches before `thebibliography` begins. The setting is local; the heading and running marks retain the surrounding direction.

`\setLTRbibitems`

Sets all bibliography entries to LTR direction.

`\setRTLbibitems`

Sets all bibliography entries to RTL direction.

`\setdefaultbibitems`

Restores the default bibliography-entry direction.

9.2 RTL Cases

`\begin{RTLcases} ... \end{RTLcases}`

Typesets two columns with a brace on the right. Cells use text mode: separate columns with &, rows with \\, and enclose mathematics in `$...$`.

```
\begin{RTLcases}
    $x$  & $x \geq 0$ \\
    $-x$ & $x < 0$
\end{RTLcases}
```

`\RTLcases{`⟨*cases*⟩`}`

In mathematics, separates rows with `\cr`. The first column uses math mode; an optional second column, introduced by &, uses text mode. Place the main expression after the argument:

```
\[\RTLcases{\text{men}\cr\text{women}}\text{people}\]
```

This example requires amsmath for `\text`.

9.3 Counter Separators

With XeTeX, `\SepMark` is provided by bidi; its counter-formatting behaviour follows that implementation.

`\SepMark{`⟨*separator*⟩`}`

Globally sets the separator in supported compound counters; the default is a period. With LaTeX and PlainTeX, the patch replaces a period immediately following the parent counter in standard sectioning formats, chapter-based equation, figure, and table numbers, and formats created by unstarred `\numberwithin`. It also sets the theorem-counter separator.

This is not a general replacement of punctuation in counter definitions. Custom formats without the expected parent-counter pattern are left unchanged. Other formatting, including Babel's localized digits, is retained.

10 Margins and Columns

10.1 Margin Notes

Use an explicit command to direct a single note. Use a switch to configure subsequent ordinary `\marginpar` calls. Text direction and margin placement are related but distinct settings.

`\LTRmarginpar[`⟨*left-text*⟩`]{`⟨*right-text*⟩`}`

Inserts a margin note with LTR text.

`\RTLmarginpar[`⟨*left-text*⟩`]{`⟨*right-text*⟩`}`

Inserts a margin note with RTL text.

The optional ⟨*left-text*⟩ argument has the same meaning as in `\marginpar`: alternative text for the left margin. Explicit commands change the note's text direction, not its placement policy.

The selected direction applies throughout a multi-paragraph note.

`\setLTRmarginpar`

Sets ordinary margin notes to LTR direction and placement.

`\setRTLmarginpar`

Sets ordinary margin notes to RTL direction and placement.

`\setdefaultmarginpar`

Restores behaviour based on the surrounding text direction.

For ordinary `\marginpar` commands, the direction switches apply when no optional left-hand text is supplied. With the optional form, the surrounding text direction is retained. The switches are local; page parity, reversed margins, and document columns still participate in placement. By default, ordinary margin notes follow the surrounding text direction.

10.2 Two-Column Output

These commands choose the placement order of completed document columns. They neither enable two-column mode nor change the direction of text within a column.

`\LTRdblcol`

Places the first column on the left.

`\RTLdblcol`

Places the first column on the right.

The default order follows the main document direction. For multi-column footnotes, see Section 8.3.

10.3 Other Column Layouts

With multicol loaded, these switches select the order of columns in subsequent `multicols` environments; they do not change the direction of the text in the columns. With XeTeX, these switches are provided by bidi; with LaTeX and PdfTeX, they are implemented by babelbidi.

`\RTLmulticolcolumns`

Places the first column on the right.

`\LTRmulticolcolumns`

Places the first column on the left.

With LaTeX or XeLaTeX and dblfnote loaded, the following switches select the order of its double-column footnotes. LaTeX uses the babelbidi implementation; XeLaTeX uses the bidi integration.

`\RTLdfnmakecol`

Places the first footnote column on the right.

`\LTRdfnmakecol`

Places the first footnote column on the left.

The initial order follows the main document direction. With XeLaTeX, bidi selects `\DFNalwaysdouble` by default; with LuaLaTeX, dblfnote's default `\DFNtrysingle` remains in effect. Use `\DFNalwaysdouble` or `\DFNtrysingle` to select the same policy explicitly on both engines.

10.4 Two-Column Table of Contents

`\twocolumnstableofcontents`

Typesets the table of contents in two columns, with a full-width section heading and `\small` entries. It takes no arguments and does not change the current language or direction. Load multicol in the preamble.

11 Language and Font APIs

These interfaces use the language roles defined in Section 5. "The selected Latin language" refers to the `latin` role, not necessarily English. Main-language font commands use Babel's main language.

11.1 Font Selection

These font commands are available with XeLaTeX and LaTeX. Optional font features use fontspec syntax. Declare the relevant Babel language before assigning its fonts.

`\settextfont[`⟨*font features*⟩`]{`⟨*font*⟩`}`

Sets the same font and font features for the `rm`, `sf`, and `tt` families of the main language.

`\setlatintextfont[`⟨*font features*⟩`]{`⟨*font*⟩`}`

Sets the same font and font features for all three font families of the selected Latin language.

These two commands apply the same font and font features to all three families (`rm`, `sf`, and `tt`). For individual language–family assignments, use `\babelfont` directly.

`\latinfont`

Selects the roman font for the selected Latin language, without changing the current language or direction. Use a group to limit its scope: `{\latinfont Latin text}`.

`\deflatinfont{`⟨*command*⟩`}[`⟨*font features*⟩`]{`⟨*font*⟩`}`

Defines an independent font-family command with `Script=Latin`. Additional font features can be specified in the optional argument. This command does not register a font with Babel or select a language.

`\setlatinsansfont[`⟨*font features*⟩`]{`⟨*font*⟩`}`

Sets the sans-serif font family for the selected Latin language.

`\setlatinmonofont[`⟨*font features*⟩`]{`⟨*font*⟩`}`

Sets the monospaced font family for the selected Latin language and configures it for use by `\latinttfamily`.

The fontspec command `\setmonofont` sets the document's monospaced font. With babelbidi, it also registers that font for the main Babel language's `tt` family, making it available to LaTeX glyph fallback.

11.2 Font Fallback

Glyph fallback supplies a character from another language's font when the current font lacks it. It is available with LaTeX and uses the fonts registered with Babel. Declare an ordered fallback chain in the preamble:

`\fontfallback{`⟨*source*⟩`}{`⟨*candidates*⟩`}`

Declares an ordered list of candidate languages whose fonts can provide missing glyphs for the source language.

```
\fontfallback{english}{persian,greek}
```

The source and candidate languages must be known to Babel, and their fonts must be declared with `\babelfont`. These declarations should be made in the preamble. The `\fontfallback` declaration may precede them; fallback mappings are resolved when the document begins.

Fallback is a font operation, not a language switch. It applies to text fonts, including custom families, but not to math fonts. It works with or without `onchar=fonts` and uses the `fontcache` setting. With a non-Latin main script, an explicit `\latinttfamily` selection uses the selected Latin language's fallback chain without changing the text language or direction.

A later declaration replaces the previous chain; an empty candidate list disables fallback:

```
\fontfallback{english}{}
```

Unknown source or candidate languages produce warnings. Repeated candidates and self-references are ignored. Fallback is not recursive and is applied per glyph before shaping. If no candidate contains a glyph, it remains in the original font. Combining-mark placement, joining, and visual consistency across fonts are not guaranteed; choose compatible fonts for the scripts involved. XeLaTeX and PdfLaTeX warn and ignore explicit fallback declarations.

11.3 Language Commands

The lowercase commands `\lr` and `\rl` select the languages assigned to the `ltr` and `rtl` roles, respectively, with their associated direction and fonts. The uppercase `\LR` and `\RL` change direction only.

`\lr{`⟨*text*⟩`}`

Typesets text using the selected LTR language.

`\rl{`⟨*text*⟩`}`

Typesets text using the selected RTL language.

`\latintoday`

Prints the current date using the selected Latin language.

`\latinttfamily`

Uses the monospaced font family of the selected Latin language without changing the active language or direction; Unicode engines only. With a Latin-script main language, it uses the current `\ttfamily` instead.

`\latinverb`⟨*delimiter*⟩⟨*text*⟩⟨*delimiter*⟩

Typesets inline verbatim text in the selected `code` language.

`\latinverb*`⟨*delimiter*⟩⟨*text*⟩⟨*delimiter*⟩

Does the same, with visible spaces.
Despite its name, `\latinverb` uses the `code` role, not the `latin` role.

11.4 Language Environments

`\begin{latin}` ... `\end{latin}`

Typesets contents in the selected Latin language.

`\begin{latin*}` ... `\end{latin*}`

Uses the same language and suppresses automatic indentation at the start of each paragraph by locally setting `\parindent` to zero.

`\begin{latinverbatim}` ... `\end{latinverbatim}`
`\begin{latinverbatim*}` ... `\end{latinverbatim*}`

The `latinverbatim` environment typesets a verbatim block in the selected `code` language. The starred form does the same, with visible spaces.
The Latin verbatim interfaces are available regardless of the main script.

11.5 Citations

`\Latincite[`⟨*note*⟩`]{`⟨*keys*⟩`}`

Works like the active `\cite`, but uses the selected Latin language and font for citation labels and their separators. An optional ⟨*note*⟩ keeps the surrounding language and text direction. With standard LaTeX citations, an Arabic comma (U+060C) precedes the note in RTL text; the surrounding font or a fallback font must provide this character. LTR text uses an ordinary comma. With cite, natbib, or biblatex, argument syntax and citation formatting follow the active `\cite`.

```
\Latincite{knuth}
\Latincite[page 23]{knuth}
\Latincite{knuth,lamport}
```

12 Engine Logos

\TeX	\LaTeX	\LaTeXe
\XeTeX	\LuaTeX	\pdfTeX
\XeLaTeX	\LuaLaTeX	\pdfLaTeX

These commands typeset the corresponding logos, preserving the surrounding language and direction. With babelbidi, they locally use English or the selected Latin language, when available.

All logos use `\xspace` for automatic spacing. Append `{}` for explicit control, as in `\LaTeX{}-based`.

The default `logostyle=auto` uses graphics-based variants when graphics or graphicx is loaded before babelbidi. Use `logostyle=simple` for variants without reflected letters. This option does not affect the standard TEX, LATEX, and LATEX 2ε constructions.

For standalone use, load babelbidi-logos directly, with `style=auto` or `style=simple`.

13 Patches

Patches are installed automatically when the relevant engine and packages are present. Load the affected packages in the preamble. The code-package integrations support loading their packages before or after babelbidi.

13.1 Common Patches

- **LATEX verbatim interfaces:** With a non-Latin main script, adjust direction and font handling for `\verb`, `\verb*`, `verbatim`, and `verbatim*`. With Unicode engines, the font patch selects `\latinttfamily` without changing the text language.
- listings: With a non-Latin main script, select the `code` language for display and inline listings, with direction handling for inline listings. The `captiondirection` key accepts `textdirection`, `LTR`, or `RTL` to control the direction of a listing caption or title independently of its code. With XETEX, this key is provided by bidi.
- minted: Register the options described in Section 14 for both Latin and non-Latin main scripts. Additional display wrappers are installed for non-Latin main scripts. For `\inputminted`, the temporary code-language switch does not write Babel language changes to the auxiliary file.
- fontspec: With Unicode engines, adapt `\setmonofont` to register the selected monospaced font for the main Babel language's `tt` family, making it available to L@TEX glyph fallback.
- Tufte and bidituftesidenote: Adapt the directional notes supplied by this package and the biditufte-book and biditufte-handout classes. For Tufte classes, margin placement uses physical page coordinates independently of note text direction. The requested margin is retained when it fits; if only the opposite margin has room for the note, that margin is used instead. This also applies to reversed margins and mirrored page geometry. The page must have enough room for a note in at least one margin. With XELATEX, the original Tufte classes also use their bidi adapter for ordinary sidenotes. With L@LATEX, provide their directional note handling; with XELATEX, retain the bidi implementation. The package's `\LTRsidenote` and `\RTLsidenote` accept optional

note numbers and vertical offsets, while `\LTRmarginnote` and `\RTLmarginnote` accept an optional vertical offset. The directional footnote commands `\LTRfootnote` and `\RTLfootnote` also place notes in the margin. With hyperref and LaTeX, labels inside numbered directional notes link to the corresponding note. With XeLaTeX, their targets follow the bidi implementation.

- **Labels and link targets:** Remove Babel counter-direction wrappers while constructing label keys and hyperlink target names, without removing them from displayed counter values.

13.2 XeLaTeX Patches

When hyperref is loaded later, avoid a recursive footnote definition in the installed bidi integration.

- tabulary: Adapt the preliminary column-width measurement in RTL tables; the final table keeps its original direction.
- natbib: Adjust citation-link handling in RTL text with recent hyperref versions.
- fontspec: Disable its automatic math-font setup when Babel bidi mode is active. This is not a general mathematics-direction patch.

13.3 LaTeX Patches

- **Boxes and tables:** Adjust `\makebox` positioning and interchange the handling of `l` and `r` alignment in language-driven RTL contexts. With the default `basic` bidi mode, explicitly directed RTL blocks retain luabidi's cell alignment. The table patch changes cell alignment, not the declared column sequence.
- **Footnotes:** Adjust paragraph direction in RTL contexts and handle footnotes issued in mathematics in RTL-main documents.
- beamer: Adjust rounded-box layout, except with beamer-rl.

For RTL-main documents, additional LaTeX patches apply:

- **TikZ:** Use LTR paragraph direction inside `tikzpicture`, except with beamer-rl.
- fancyhdr: Set `\headheight` to `1.2em` and wrap headers and footers in LTR runs. This value is not an automatic measurement of the header contents.

14 Minted Package

Load minted in the preamble and satisfy the external-tool requirements documented in its manual:

```
\usepackage{minted}
```

The options in this section are extensions supplied by babelbidi, not standard minted options. They are available with both Latin and non-Latin main scripts. Select the language and direction separately from the syntax-highlighting lexer.

14.1 Unicode TeX Commands

Pygments' standard `latex` lexer recognizes only the first character of a non-Latin command name. With XeLaTeX or LuaLaTeX, BabelBidi adjusts the displayed highlighting of such command names in `minted` code blocks and file listings.

This adjustment, controlled by the package option described in Section 4.4, does not replace Pygments' lexer and does not affect inline listings or commands inside math mode. When Pygments splits a name into several syntax tokens, for example around an underscore, the remainder may retain its original highlighting.

```
\begin{minted}{latex}
    \نمونه{متن فارسی}
\end{minted}
```

14.2 Language and Direction

`textlang`=⟨*language*⟩

Selects an already declared Babel language. An empty value uses the language assigned to the `code` role.

`textdir`=⟨*direction*⟩

Accepts `ltr` or `rtl`. It is applied after language selection, so an explicit direction overrides the direction established by `textlang`.

```
\begin{minted}[textlang=persian,textdir=rtl]{text}
    متن فارسی
\end{minted}
```

The language `persian` must already be declared and have a suitable font. The argument `text` above is the minted lexer, not the Babel language. Defaults can also be set with `\setminted`:

```
\setminted{textlang=english,textdir=ltr}
```

14.3 Character Directions

With LuaLaTeX, the following option temporarily changes Babel character-direction properties:

`chardir`=⟨*pairs*⟩

Accepts a comma-separated list of `{character}{direction}` pairs. Direction values include `l`, `r`, `al`, `en`, `an`, `on`, `et`, and `cs`.

For example, assign LTR direction to a backslash and closing brace:

```
\begin{minted}[textdir=rtl,
    chardir={{\\}{l},{\}}{l}}]{latex}
    این روش در
    \lr{Chapter 2 (Examples)}
    توضیح داده شده است.
\end{minted}
```

Saved properties are restored after a `minted` environment or `\inputminted`. Do not use nonempty `chardir` with `\mintinline`: Babel requires character-property changes in vertical mode. A nonempty `chardir` value raises an error with XeLaTeX or PdfLaTeX; it is not silently ignored.

14.4 Layout

Inside `minted` environments, line numbers use Latin digits. With LaTeX, frame lines and padding receive LTR geometry adjustments. For a non-Latin main script, the additional LaTeX display wrapper uses a minipage. Such wrapped displays cannot break across pages; keep this in mind for long listings.

15 Limitations and Compatibility

- Interface compatibility does not guarantee identical output across engines. The XeTeX implementations also depend on the installed bidi version.
- Compatibility has been tested with `dblfnote.sty` version 1.1 (2018/09/03), from the yafoot 1.1 bundle. Other versions have not been independently tested.
- Direction control does not supply language data, input or font encodings, fonts, or missing glyphs. Some patches rely on Babel or other packages' internals. Inspect the resulting PDF, particularly when other packages also modify layout; a successful compilation is not a bidirectional-layout check.
- Extra footnote layouts modify insertion and output handling. Check their compatibility with classes or packages that modify these mechanisms, including the position and order of notes. See Section 8.3 for the engine-specific behaviour and restrictions.

16 License

The babelbidi package is released under the LaTeX Project Public License (LPPL), version 1.3c or later. The full license text is available in the `LICENSE` file. This independent guide adapts presentation code from the Babel manual. Source and copyright notices for this and other reused material are in `THIRD-PARTY-NOTICES.md`.

17 Author and Maintainer

The package is developed and maintained by **Amer Amikhteh**.

Source code, development versions, and the issue tracker are available in the project repository:

`https://github.com/gambi-shah/babelbidi`

The repository also contains test documents in the `test/` directory. These development files are not included in the CTAN distribution.

Please report bugs on the project's GitHub issue tracker and include:

1. A small, complete example that reproduces the problem.
2. The babelbidi and Babel versions, and the TeX engine used (with its version, if known).
3. A brief description of the expected and actual results, with any error message or a screenshot of the affected output.