emo: emoji for all
(LaTeX engines)

Robert Grimm

Version v0.1 (2023/03/07)

Abstract

Emo implements the `\emo{`⟨*emoji-name*⟩`}` command for including color emoji such as 🏝️ or 🦜 in your documents independent of input encoding or LaTeX engine. The implementation uses the Noto color emoji font if the engine supports it and includes PDF graphics otherwise. The latter are derived from Noto's SVG sources, so the visual appearance is very similar. The source repository is at `https://github.com/apparebit/emo`.

Contents

1 Installation

To install the emo package, first extract embedded files from `emo.dtx` by executing:

```
$ pdflatex emo.dtx
```

Next, building emo's documentation with indices requires executing:

```
$ source build.sh
```

Finally, actually using emo requires that you put the following files somewhere LaTeX can find them. While your current project's directory will do in a pinch, following the TeX Directory Structure avoids clutter and ensures that any TeX tool can find emo's files when needed. They are:

1. `emo.sty` with package implementation;

2. `emo.def` with emoji table;
3. `NotoSerifTC-Regular.otf` for `\lingchi`;
4. *all* PDF files in the `emo-graphics` directory.

When running on the LuaLaTeX engine, the emo package also uses the Noto color emoji and Linux Libertine fonts, with the latter use to render `\YHWH`. Neither file is included with emo's distribution, since both of them are distributed with major TeX distributions already.

If you want to change the inventory of supported emoji, execute:

```
$ python3 scripts/emo.py
```

2 Usage

As usual, you declare your document's dependency on emo with `\usepackage{emo}`. In addition to the unadorned form, emo takes up to two options:

extra Also define the `\lingchi` and `\YHWH` macros, which produce 凌遲 and יהוה, respectively, and are documented below.

index Create an emoji index tagged `emo` with the `.edx` extension for the raw index and the `.end` extension for the processed index. This option relies on the `index` package, generates the raw `.edx` file, but does not build or use the processed index.

`\emo` `\emo{`⟨*emoji-name*⟩`}` expands to the named emoji. For LuaLaTeX, it uses the Noto color emoji font. For all other engines, it uses PDF graphics. That way, `\emo{desert-island}` results in 🏝 and `\emo{parrot}` results in 🦜.

Since LaTeX tends to produce a lot of command line noise about underfull boxes and loaded fonts, it's a easy to miss meaningful warnings. For that reason, `\emo` expands to an attention-seeking error message upon undefined emoji names. For example, `\emo{boo}` produces Bad `\emo{boo}`.

`\lingchi` `\YHWH` The `\lingchi` and `\YHWH` macros take no arguments and produce 凌遲 and יהוה, respectively. They are only available if emo is used with the extra option. The first command produces is the Chinese term for "death by a thousand cuts." While originally an execution method, the term applies to surprisingly many software systems as well. The second command produces the Tetragrammaton, the Hebrew name for God. Observant Jews do not speak the name, not even in thoughts. In my mind, that nicely mirrors the very incomprehensibility of יהוה. Both macros preserve a subsequent space as space, no backslash needed.

3 Implementation

Let's start the package implementation:

```
⟨*package⟩
```

Except, the package implementation started near the top of the emo.dtx file, so that version number and date are more visible and declared only once. But that's also well before the documentation preamble and hence cannot be included in the annotated implementation. Nonetheless, we can simulate the lines:

```
\NeedsTeXFormat{LaTeX2e}
\ProvidesPackage{emo}[2023/03/07 v0.1 emo for all]
```

And no, I didn't repeat the version number and date. Check emo.dtx.

3.1 Package Options

Emo's extra and index options are simple flags. So, we declare a \newif for each option and, if the package use includes an option, it just toggles the conditional's state:

```
\newif\ifemo@extra\emo@extrafalse
\DeclareOption{extra}{\emo@extratrue}
\newif\ifemo@indexing\emo@indexingfalse
\DeclareOption{index}{\emo@indexingtrue}
\ProcessOptions\relax
```

3.2 Dependencies

The dependency on inputenc effectively declares this file's encoding to be UTF-8. The XeTeX and LuaTeX engines already expect files to be encoded that way and hence ignore the declaration. However, pdfTeX supports other (legacy) encodings and needs to be told.

```
\RequirePackage[utf8]{inputenc}
```

Depending on TeX engine, this package requires either fontspec or graphicx as the emoji-emitting backend. In turn, to tell the engines apart, it requires iftex.

```
\RequirePackage{iftex}
\ifluatex
\RequirePackage{fontspec}
\else
\RequirePackage{graphicx}
\fi
```

Emo requires xcolor for formatting highly visible error messages within the text. Always including another package that is only used when there are errors is not ideal. But when I tried calling \RequirePackage for xcolor from inside the error macro, it didn't work. Alternatively, I could make in-text errors optional.

```
\RequirePackage{xcolor}
```

Finally, emo's options also have dependencies, with extra requiring the xspace package and index requiring the index package:

```
\ifemo@extra
\RequirePackage{xspace}
\fi
\ifemo@indexing
\RequirePackage{index}
\fi
```

3.3 The Emoji Table

For each emoji with a PDF graphic in the `emo-graphics` directory, a macro named `\emo@emoji@`⟨*emoji-name*⟩ expands to its Unicode sequence. With over 3,000 distinct emoji in Unicode 15, emo relies on a Python script for populating the graphics directory and writing the table to the `emo.def` file. Since the package code does not change after installation but the emoji table may very well change, they are kept separate for now. Alternatively, we could use DocStrip to assemble the package file from three parts, the code from the previous sections, then the contents of the emoji table in `emo.def`, and then all subsequent code.

```
\input{emo.def}
```

3.4 Internal Macros

`emo@error@fg`
`emo@error@bg`
`emo@error`

Define two colors and a function that uses the two colors for formatting an attention-grabbing error message. If you use an invalid emoji name and overlook the warning in the console, you *will* notice the error messsage in the document thusly formatted.

```
\definecolor{emo@error@fg}{rgb}{1,1,1}
\definecolor{emo@error@bg}{rgb}{.6824,.0863,.0863}
\def\emo@error#1{%
    \colorbox{emo@error@bg}{%
        \textcolor{emo@error@fg}{%
            \textsf{Bad} \texttt{\textbackslash emo\{#1\}}%
        }%
    }%
}
```

`emo@ifdef`

Validate the emoji name given as first argument. The macro expands to the second argument if the name is valid and an error message otherwise. Its implementation relies on the `emo@emoji` table.

```
\def\emo@ifdef#1#2{%
    \ifcsname emo@emoji@#1\endcsname#2\else%
        \PackageWarning{emo}{Unknown emoji name in ‘\string\emo{#1}’}%
        \emo@error{#1}%
    \fi%
}
```

`emo@index`

If indexing is enabled, record the use of an emoji. Otherwise, do nothing.

```
\ifemo@indexing
\newindex{emo}{edx}{end}{Emoji Index}
\def\emo@index#1{\index[emo]{#1}}
\else
\def\emo@index#1{}
\fi
```

3.5 User Macros

emo

Emit the named color emoji. Both the font-based version for LuaTeX and the graphics-based fallback validate the emoji name and then invoke the \emo@index macro. But they differ in how they actually display the emoji. The LuaTeX version turns the emoji name into its Unicode sequence and wraps that in a group that also uses the previously declared Noto color emoji font. The fallback version instead includes a suitably sized PDF graphic.

```
\ifluatex
\newfontface\emo@font[Renderer=Harfbuzz]{NotoColorEmoji.ttf}
\newcommand\emo[1]{%
    \emo@ifdef{#1}{%
        \emo@index{#1}%
        {\emo@font\csname emo@emoji@#1\endcsname}%
    }%
}
\else
\newcommand\emo[1]{%
    \emo@ifdef{#1}{%
        \emo@index{#1}%
     \raisebox{-0.2ex}{\includegraphics[height=1em]{./emo-graphics/#1}}%
    }%
}
\fi
```

lingchi
YHWH

The definitions for the optional \lingchi and \YHWH macros follow from that of \emo, except that (a) there are no arguments to validate and hence no equivalent to \emo@ifdef; (b) Hebrew is written right-to-left and hence \YHWH requires a \textdir TRT; (c) subsequent space should be preserved and hence both macros end with \xspace. While it would be nice to use Unicode inside the groups for the LuaTeX macros, doing so breaks the package documentation. So \char it is.

```
\ifemo@extra
\ifluatex
\newfontface\emo@chinese{NotoSerifTC-Regular.otf}
\newfontface\emo@hebrew{LinLibertine_R.otf}
\newcommand\lingchi{%
    \emo@index{lingchi}%
    \begingroup\emo@chinese \char"51CC\char"9072\endgroup%
    \xspace}
\newcommand\YHWH{%
68    \emo@index{YHWH}%
69  \begingroup\textdir TRT\emo@hebrew \char"5D9\char"5D4\char"5D5\char"5D4\endgroup%
70    \xspace}
71 \else
72 \newcommand\lingchi{%
73    \emo@index{lingchi}%
74  \raisebox{-0.2ex}{\includegraphics[height=1em]{./emo-graphics/lingchi}}%
75    \xspace}
76 \newcommand\YHWH{%
77    \emo@index{YHWH}%
78  \raisebox{-0.2ex}{\includegraphics[height=1em]{./emo-graphics/YHWH}}%
79    \xspace}
80 \fi
81 \fi
```

Et voilà. That's it!

```
82 ⟨/package⟩
```

Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.