Minimal Manuals

Manual

Simple and sober manuals inspired by the OG linux manpages.

min-manual 0.1.0 MIT

Maycon F. Melo[*]

QUICK START

```
#import "@preview/min-manual:0.1.0": manual
#show: manual.with(
  title: "Longer, Readable Name",
  description: "Short description, generally two or less lines.",
  authors: "You Name <https://your-url.com>",
  cmd: "short-name",
  version: "0.1.0",
  license: "MIT",
  logo: image("logo.png")
)
```

DESCRIPTION

Generate modern manuals, without loosing the simplicity and looks of old manuals. Instead of the equally nostalgic LaTeX manuals, this package draws inspiration from the Linux manpages, as they look in terminal emulators until today, and adapts it to the contemporary formatting possibilities.

This package is designed to universally document any type of program, or code including Typst packages and templates — this very manual was written using this same package, so reading it you may see how it looks and reading its source you may see how it is written.

This manual will be updated only when new versions break or modify something; otherwise, it will be valid to all newer versions starting by the one documented here.

[*] https://github.com/mayconfmelo

OPTIONS

Those are the full list of options available and its default values:

```
#import "@local/min-manual:0.1.0": manual
#show: manual.with(
  title: none,
  description: none,
  authors: none,
  cmd: none,
  version: none,
  license: none,
  logo: none,
  manual-author: none,
  toc: false,
  paper: "a4",
  lang: "en",
  justify: true,
  line-space: 0.65em,
  par-margin: 1.2em,
  margin: 1in,
  font: "Arial",
  font-size: 12pt
)
```

Seems like an awfull lot to start with, but let's just break down all this to understand it better, shall we?

`title:` string content (*required*)

The manual title; generally the longer, more readable name of what is been documented, or its name itself otherwise. If not set, the compilation fails with an error.

`description:` string content

A short description of what is being documented; generally two lines long or less.

`authors:` string array (*required*)

The author or authors of what is being documented — not the manual. When more than one author, each string can be `"AUTHOR-NAME <AUTHOR-URL>"`, where `<AUTHOR-URL>` is a optional URL associated to the author, like Github profile or webpage. If ot set, the compilation fails with an error.

`cmd:` string content

The command or code name used to invoke what is being documented.

`version:` string content

The version of what is being documented. Usefull when different versions have diferent behaviors.

`license:` string content (*required*)

The license of what is being documented — notnthe manual.

`logo:` image

The logo image of what is being documented.

`manual-author:` string

The author of the manual — not what is being documented. If not set, fallback to the first (or only) one of `authors`.

`toc:` boolean

Defines whether the manual will have a table of contents or not.

`paper:` string

Defines the page paper type — and its size therefore.

`lang:` string

Defines the language of the written text.

`justify:` boolean

Defines if thebtext will have justified aligment.

`line-space:` length

Defines the space between lines in the document.

`par-margin:` length

Defines the margin space after each paragraph. Set it tue same as `line-space` to remove get paragraphs without additional space in between.

`margin:` length

Defines the document margins.

`font:` string array

Defines the fint families used for the text. If set multiple fonts, will try to aplly the first one, and fallback to the next one if not found. If some gliph in the text does not exist in the actual font, it fallbacks to the next font too.

font-size: length

Defines the size of the text in tue document.

ARGUMENT COMMAND

```
#import "@local/min-manual:0.1.0": arg
#arg(
  name, types,
  required: false
)[
  body
]
```

This command offers a convenient way to document the arguments — or parameters, or options, or whatever they are called.

name string raw (*required*)

The argument name; can have sintax highlight if set as a raw text, like ```LANG NAME```, where LANG is the programing language name.

types string array none

The value types the argument takes.

body content (*required*)

A brief description of what the argument does.

PACKAGE CITATION COMMAND

```
#import "@local/min-manual:0.1.0": pkg
#pkg(
  name,
  url
)
```

As a convenient bonus, the Typst Universe, Python Pip/Pypi, Rust crates, and GitHub repositories can be cited in an even easier way:

```
#import "@local/min-manual:0.1.0": univ, pip, crate, gh
#univ(name)
#pip(name)
#crate(name)
#gh(name, user)
```

Thse are small helper commands that simplifies the citation of any type of external package, crate, or library using its repository URL. They just insert a link in the package name for convenient access, and creates an additional footnote to it, for usability in eventual printed manuals.

`name` string (*required*)

The name of the package, or library, or crate, or anything else, as it appears in the package repository, e.g.: just the `babel` of `https://ctan.org/pkg/babel`.

`url` string (*required*)

The package repository URL without package name path, e.g.: just the `https://ctan.org/pkg/` of `https://ctan.org/pkg/babel`. Used by `#pkg`.

`user` string (*required*)

The GitHub user, as it appears in GitHub repositories, e.g.: just the `packages` of `https://github.com/typst/packages`. Used by `#gh`.

COPYRIGHT

The logo was obtained from Flaticon website.