

---

# pandocgen

This is a generation framework for yielding PDF and HTML from good old Markdown files. It uses the eminent [Pandoc tool](#), so the Markdown files can use Pandoc extensions to provide a slicker output, including mathematical expressions. For the pure Markdown syntax and its semantics, there is a good introduction on [Daring Fireball](#).

**NOTE:** for some reason, the creator of Pandoc, John MacFarlane, has not named the Pandoc Markdown extension language. I sometimes refer to it as **PandocMarkdown**.

**NOTE:** Pandoc is capable of translating between a host of formats, but this **pandocgen** project focuses on (Pandoc-)Markdown input. See the graph at the bottom of this document for all the various conversion options of Pandoc; it is quite mind-blowing. The image is from the Pandoc site and is copyrighted by John MacFarlane.

## 1.1 Very, Very Quick Intro...

1. Add this project as a submodule

```
git submodule add git@github.com:davber/pandocgen.git
```

2. Create some beautiful Markdown file, `MyCoolDoc.md`

3. Create a Makefile like this:

```
BASE=MyCoolDoc
include pandocgen/Makefile
```

4. Make space for the generated files:

```
mkdir gen
```

5. Create the PDF and HTML files:

```
make all
```

That is it! Now you have a PDF and HTML version of your Markdown document.

For a quick generation, you can actually generate this README file — people always enjoy others eating their own dog food (or actually others eating pet food in general...) — by

```
make --file sample.mk
```

where `sample.mk` is this short make file

```
BASE=README
RES_OUT=rez/diagram.png
include Makefile
```

This will generate output files in the `gen` directory. They are also uploaded to this Wiki, as a [PDF file](#) and [HTML file](#).

## 1.2 Dependencies

There are some dependencies, though:

1. A Gnu **make** command, preferably version 3.80 or later. On most decent machines, this is already installed.
2. LaTeX, such as [TeX Live](#).
3. [Pandoc](#) — the tool actually doing the generation

## 1.3 Using It...

To use this framework, there are two paths:

1. Include the provided Makefile in your own Makefile, after a section where you specify a few custom parameters, as defined in the Custom Parameters section.
2. Set the custom parameters as environment variables and use the provided Makefile as is.

## 1.4 Building Targets

All target versions of the Markdown document are generated in a `gen` directory relative the current working directory. As a convenience, this project contains such a directory in case you are running `make` from there.

Each of the target formats has a corresponding `make` target, so you can issue one of:

```
make pdf
make html
make tex
```

There is also a universal target, which builds all formats:

```
make all
```

## 1.5 Customer Parameters

These are the parameters you can set – either via a initial section in an embedding Makefile or as environment variables:

- `BASE` - that is the name of the Markdown document, **without** suffix, such as `MyCoolDoc`, which will then generate `MyCoolDoc.pdf`, `MyCoolDoc.html` and `MyCoolDoc.tex` in the `gen` directory. **MANDATORY**
- `RES_IN` - the input files for resources, such as images, needed by the document. This often includes Graphviz Dot files or other input formats for PDF- and PNG-based images. **OPTIONAL**
- `RES_OUT` - the corresponding generated resource files, which are often PDF and PNG files. **OPTIONAL**.
- `RES_GEN` - the full command to generate the `RES_OUT` files from the `RES_IN` files. **OPTIONAL**

**NOTE:** so there is only **one** mandatory parameter to set, and that is `BASE`.

**NOTE:** the default behavior, described above, actually allows you to include resources in an input-ready format, such as raw PNG and PDF files, by merely setting the `RES_OUT` to those files and let the other two resource-related parameters be. That will translate into a no-op for that `make` step.

## 1.6 Helper Files

The helper files reside in the `input` directory.

The helper files are:

- `my-template.latex` - this is the main template for LaTeX generation and, indirectly, for PDF generation. It uses some parameters that can be set from command line — and is actually set by the provided Makefile — such as `documntclass`, which the Makefile sets to `memoir`. See the Makefile for some of those parameters used.
- `mytitle.tex` - this is the template for the title page, for LaTeX (and PDF...)
- `mychapter.tex` - specifies the look of chapter headings for LaTeX (and PDF...)
- `macros.tex` - some TeX macros. **NOTE** these macros are actually expanded by Pandoc itself in the case of non-TeX-based generation — such as HTML — so one can have shortcuts or other macros even for HTML.

## 1.7 The Completely Connected World Of Pandoc

Can you count the number of translations possible? ...

Again: Image copyright John MacFarlane

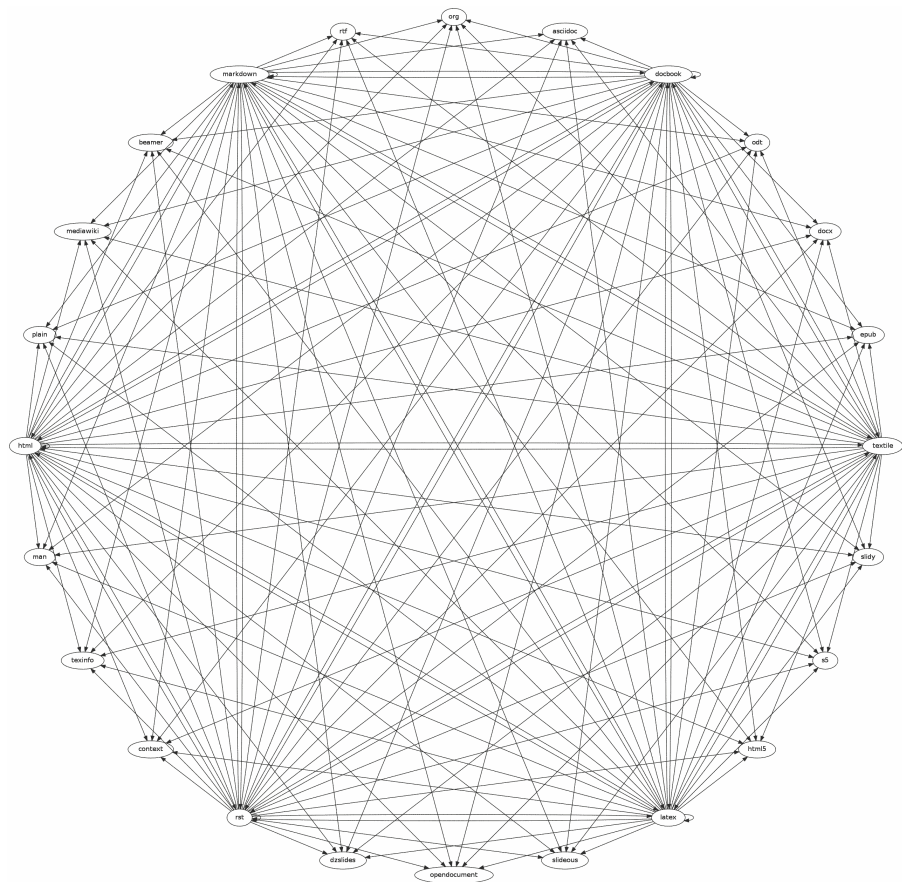


Figure 1.1: Pandoc Format Conversions