Table of Contents

1. BibTeX bibliography

`prodockit.bibliography` creates citations and a formatted reference list from a `.bib` file. Use it when you have many sources or need a particular citation style, such as Harvard, APA, IEEE, or Vancouver.

For a short reference list that you prefer to write yourself, use Hand-written citations and references.

1.1 Before you start

Pandoc must be installed before you use this extension:

macOS

```
brew install pandoc
```

Windows

```
winget install --source winget --exact --id JohnMacFarlane.Pandoc
```

Linux (Ubuntu)

```
sudo apt update
sudo apt install pandoc
```

Check the installation:

```
pandoc --version
```

See Pandoc's installation guide for its installers and other operating systems.

1.2 Enable the extension

Enable it in `zensical.toml` and set `bib_file` to the name of your bibliography file:

```
[project.markdown_extensions."prodockit.bibliography"]
bib_file = "references.bib"
```

1.3 Add and cite a source

Create `references.bib` in the project root:

references.bib

```
@book{chacon2014,
  author    = {Chacon, Scott and Straub, Ben},
  title     = {Pro Git},
  edition   = {2},
  year      = {2014},
  publisher = {Apress}
}
```

Create a references page and put the bibliography marker alone on its own line:

docs/references.md

```
# References

\bibliography
```

Now cite the entry from any page with `\cite{id}`:

Markdown

```
Git is a distributed version control system \cite{chacon2014}.
```

Result (default style)

Git is a distributed version control system (Chacon and Straub 2014).

The exact punctuation and ordering come from the selected citation style; see Choosing a citation style.

The citation links to the matching entry on the References page. Keep the `\bibliography` marker on that page so the reference list is created.

1.4 Configure the reference list

1.4.1 Choose the bibliography settings

These are the usual project settings:

```
[project.markdown_extensions."prodockit.bibliography"]
bib_file = "references.bib"
csl_style = "harvard-cite-them-right.csl"
unresolved = "?"
```

Setting	Default	What it controls
`bib_file`	`"references.bib"`	Path to the `.bib` file, relative to the directory where you run `zensical build` or `zensical serve`.
`csl_style`	`""` (Pandoc's default)	Path to the `.csl` file that controls citation and reference-list formatting. Leave it out to use Pandoc's default style.
`unresolved`	`"?"`	Text shown for a citation key that cannot be found in the `.bib` file.
`source`	`""` (detected automatically)	Advanced: identifies the current page when using the extension outside Zensical. Leave it unset in `zensical.toml`.

Only `bib_file` is normally required. Add `csl_style` when you need a particular citation style, and change `unresolved` only when you want another missing-citation marker.

Put a bare `\bibliography` marker, alone on its own paragraph, wherever you want the complete, formatted reference list to appear - typically a dedicated References page, kept at the end of `nav` as an appendix, the same convention `prodockit.citations`' own hand-authored references pages already use:

The quick start above uses the default form, `\bibliography`.

Every entry in `bib_file` appears, in the order your chosen CSL style sorts them (alphabetically by default) - not just the ones actually cited, the same way LaTeX's `\nocite{*}` includes every `.bib` entry regardless of whether it's cited in the document. A `\cite{id}` written on any page links directly to its own entry here, adjusted for that page's own directory depth - a real Zensical clean-URL link like `prodockit.refs` / `prodockit.citations` already build, not a bare `#id` fragment that would 404 from a different page.

`\bibliography` takes two optional parameters narrowing this down to just what's actually cited, and/or a different `.bib` file than the configured default - see Multiple sections: References and Bibliography below.

1.4.2 Multiple sections: References and Bibliography

A style guide often distinguishes two different lists:

- **References** (or "Works Cited") - a strict list of only the sources actually cited in the text.
- **Bibliography** - a broader list: everything cited, *plus* background reading the author wants to list even though it's never individually cited inline.

`\bibliography` takes two optional, positional parameters for exactly this:

```
\bibliography{<file>}{<true|false>}
```

- `<file>` - which `.bib` file *this* marker draws from. Leave it empty (`{}`) or omit it entirely to use the configured `bib_file`.
- `<true|false>` - `true` restricts this marker's list to only entries actually `\cite{}`-cited somewhere in the build; `false` (the default, and the only behaviour before these parameters existed) keeps every entry, cited or not.

Write two markers to get both sections from the same `.bib` file:

```
<!-- references.md -->
# References

\bibliography{}{true}
```

```
<!-- bibliography.md -->
# Bibliography

\bibliography{}{false}
```

or from two different files, if background/further-reading sources live separately from the ones actually cited:

```
<!-- references.md -->
# References

\bibliography{references.bib}{true}
```

```
<!-- bibliography.md -->
# Bibliography

\bibliography{background.bib}
```

A `\cite{id}` links to whichever marker's page actually lists that entry, based on which `.bib` file defines it - in the common single-file case (one bare `\bibliography`, as in Quick start above), that's still just the one page, exactly as before. `csl_style` stays a single, extension-wide setting - only the file and cited-only flag are per-marker.

1.4.3 Choosing a citation style

Point `csl_style` at any `.csl` file - your institution's own house style, or one of the thousands available from the Citation Style Language project (the same repository Zotero/Mendeley pull styles from). This project's own docs, and `prodockit-template` / `prodockit-userguide` 's hand-authored references, already follow Cite Them Right Harvard - `harvard-cite-them-right.csl` reproduces that exact style automatically:

```
[project.markdown_extensions."prodockit.bibliography"]
bib_file = "references.bib"
csl_style = "harvard-cite-them-right.csl"
```

renders (confirmed directly against the same `.bib` file used above):

```
<p>Git is a distributed version control system <span
class="citation">(Chacon and Straub, 2014)</span>.</p>
...
<div id="ref-chacon2014" class="csl-entry">
Chacon, S. and Straub, B. (2014) <em>Pro git</em>. 2nd edn. New
York: Apress. Available at: <a href="https://git-scm.com/
book">https://git-scm.com/book</a>.
</div>
```

- exactly the format already hand-typed throughout this project's own reference lists, just generated instead. Leaving `csl_style` unset uses Pandoc's own default (a Chicago author-date style) instead. Confirmed directly: the exact same `.bib` file, with only `csl_style` changed, produces correctly (and very differently) formatted output - author-date parenthetical

citations and a hanging-indent reference list for APA, numbered `[1]` citations and a numbered list for IEEE, and so on - with no other configuration.

1.4.4 Unresolved citations

A key that doesn't resolve to a `.bib` entry renders the `unresolved` marker (`?` by default), unlinked:

```
\cite{does-not-exist}
```

renders `?`, with no link.

1.5 Reference

1.5.1 Syntax

Syntax	Purpose
`\cite{<id>}`	Cite one entry from a configured `.bib` file
`\bibliography`	Generate every entry from the configured `bib_file`
`\bibliography{<file>}`	Generate every entry from another `.bib` file
`\bibliography{<file>}{true}`	Generate only entries cited in the build
`\bibliography{<file>}{false}`	Generate every entry, cited or not

Only a single key is supported - unlike `prodockit.citations`' `\citeref{id1,id2,...}`, a multi-key citation isn't matched by this extension's own syntax at all (falls through as literal text, a visible, honest "not supported" rather than a silently wrong result) - see Comparing the two approaches for why.

Like prodockit.citations, `\cite{...}` is recognised the same way Python-Markdown's own inline syntax is, so it's protected inside inline code spans and fenced code blocks.

`<file>` and `<true|false>` are both optional - see Multiple sections: References and Bibliography above for what they do and when to use them. Put the marker alone on its own paragraph/line.

The extension delegates CSL formatting to Pandoc. Contributors changing that integration should read Extension integration.

1.6 Comparing the two approaches

Both extensions solve the same problem - cite a source by key, get a formatted reference list - but make a fundamentally different tradeoff about where the formatted text comes from. They can be enabled together in the same build without conflict (this project's own docs do, to demonstrate both side by side), though a typical single project only needs one.

	prodockit.citations	prodockit.bibliography
Source of truth	A hand-typed paragraph, once, tagged `data-cite-text`	A `.bib` file entry
Reference list	You write it, by hand, in full	Generated automatically
Citation style	Whatever you typed - one style, fixed	Any CSL style, swappable via one setting
Multi-key citations (`\citeref{a,b}`)	Yes - each key individually linked	Not supported (falls through as literal text)
External dependencies	None	`pandoc` on `PATH`, even without a PDF build
Editing a reference	Edit the prose by hand, on the references page	Edit the `.bib` entry once, everywhere it's cited updates
Separate References/ Bibliography sections	Not built in - would need two hand-authored lists kept in sync manually	Built in - `\bibliography{<file>}{<true|false>}` generates a strict cited-only list and/or a broader everything-included list, see Multiple sections

Where `prodockit.citations` fits best: a short reference list, a house style unlikely to ever change, or a project that doesn't want a `pandoc` dependency for its website build at all (only for its optional PDF, via prodockit.pdf, which already needs `pandoc` anyway).

Where `prodockit.bibliography` fits best: a longer, actively-maintained reference list; needing to match a specific institution's CSL style (or switch between several, e.g. a thesis needing IEEE for one chapter's publications list and Harvard for the rest of the document - one `.bib` file, several instances with different `csl_style` values); or wanting a citation added anywhere in the document to also add it to the reference list, correctly formatted, with no hand-typing at all.

1.6.1 What this project's own template and user guide currently do

prodockit-template has adopted `prodockit.bibliography`, and is a worked example of Multiple sections in a real project: a `references.md` page (`\bibliography{}{true}`) listing only the sources actually `\cite{}`-cited in the document, from a `references.bib` file, and a separate `bibliography.md` page (`\bibliography{bibliography.bib}`) listing everything in a distinct `bibliography.bib` - further reading the author wants to list even though it's never individually cited inline. Both pages share one `csl_style` (Cite Them Right Harvard), configured once on the extension.

prodockit-userguide still uses `prodockit.citations`' hand-authored approach for its own reference list - a short, relatively static one (around ten entries) - and enables `prodockit.bibliography` alongside it only to demonstrate the two coexisting without conflict, the same way this project's own docs do, not as its real reference mechanism. A project outgrowing a short, hand-typed list - a longer, frequently-updated bibliography, or needing to match a specific CSL style - is exactly the case `prodockit.bibliography` is built for, as prodockit-template's own adoption shows.

> **A newer authoring option**
>
> `prodockit.bibliography` is newer and less battle-tested than `prodockit.citations`. The project-wide maturity and compatibility boundary is documented under Support and compatibility.

1.7 Customise with a CSS style sheet

Element	Condition	Hook
`<span>` wrapping a resolved `\cite{id}`	always	`class="prodockit-bib-cite"`
`<span>` wrapping an unresolved `\cite{id}`	always	`class="prodockit-bib-cite prodockit-bib-cite-unresolved"`
Each generated reference-list entry	always	`class="csl-entry reference"`

Every generated reference-list entry also gets `class="reference"` (in addition to Pandoc's own `csl-entry`) - matching the class `prodockit.citations`' own hand-authored entries already use, so `prodockit.zensical_macros`' `reference_style()` / prodockit.pdf's own `reference_style` setting apply uniformly, whether an entry was hand-typed or generated.

Index

P