

Table of Contents

- 1. About prodockit 2
 - 1.1 Understand the foundation 2
 - 1.2 Know the project family 2
 - 1.3 Choose where to continue 2

1. About prodockit

This section is for anyone evaluating or using prodockit who needs to understand its scope, dependencies, support, release history, or legal terms.

prodockit is a Python package for professional and academic documentation built with [Zensical](#). It adds Markdown extensions, website macros, PDF generation, output checks, and project-management commands without requiring a document author to build those pieces independently.

1.1 Understand the foundation

prodockit builds on Python-Markdown, Zensical, and [PyMdown Extensions](#). PyMdown Blocks is a direct foundation, not merely a compatible syntax: `prodockit.steps` and `prodockit.tree` are implemented with its Blocks API and use the same slash-fenced container model as PyMdown's own blocks. Installing prodockit therefore installs `pymdown-extensions` too.

This relationship lets prodockit add specialised document blocks while retaining the nesting, configuration, and rendering conventions familiar to Zensical authors. Start with [Numbered steps](#) or [Directory trees](#) to see the shared model in use.

1.2 Know the project family

[Table 53.1](#) distinguishes the role of each repository and published document.

Table 1.1. Know the project family

Project	Role
prodockit-extensions	The Python package, command-line tools, implementation, tests, and this technical reference
prodockit-template	The maintained starting project, including annotated GitHub and GitLab publishing automation
prodockit-userguide	The task-based guide for people creating and publishing documents with the template

The project roles in [Table 53.1](#) separate the package, its starter project, and its longer guide. The template is maintained on GitHub and synchronised to the University of Surrey's GitLab for student use. Projects created from it use the same prodockit package documented here. A downstream release is only complete when the mirror's `main` branch and release tag are both current; the maintainer procedure is documented in [Complete each downstream mirror](#).

1.3 Choose where to continue

Use [Table 53.2](#) to select the next page for the task you have in mind.

Table 1.2. Choose where to continue

Page	Use it for
Get started	Install prodockit and build a first local site
Authoring reference	Add document features, macros, PDFs, and command-line tools
Publish a document	Update and deploy a website and its outputs
Support and compatibility	Check maturity, supported versions, platforms, and known constraints
Release notes	See the implemented capability summary and short, user-relevant upgrade notes
Licence	Read the MIT License governing use, modification, and redistribution

Repository maintainers should use [Maintain prodockit](#). Developers changing the package itself should use [Contributor internals](#).