# Doc Packager

# Doc Packager

**Doc Packager** is a tool that enables packaging both evergreen and living documentation into human-readable packages.
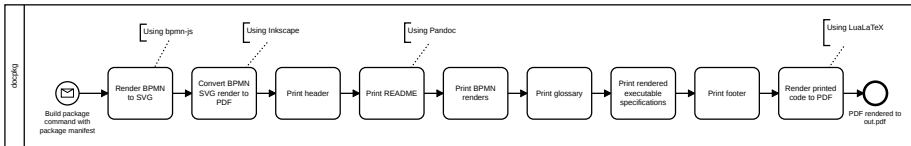
Make sure to `npm install` first.

How it works:

1. Define your project in a `README.md` file and in Gherkin executable specifications.
2. Set the documentation configurations in your package manifest file to point to the output of previous step.
3. Run `clojure -X:main` to process the output into a PDF file.

In the current example project, steps 3 and 4 are automated in GitHub Actions. This way, each commit leads to a new PDF explaining the project in the Package docs workflow history.

It is currently an experiment, not useable in production. Contact [@sander](https://github.com/sander) if you have questions or want to collaborate.

# Business processes

## Building a package



Build package command with package manifest → Render BPMN to SVG (Using bpmn-js) → Convert BPMN SVG render to PDF (Using Inkscape) → Print header → Print README (Using Pandoc) → Print BPMN renders → Print glossary → Print rendered executable specifications → Print footer → Render printed code to PDF (Using LuaLaTeX) → PDF rendered to out.pdf

## Glossary

**Documentation package:** A curated collection of rendered documents from one or more repositories.

**Documentation package manifest:** A document listing the source documents to be processed into a documentation package.

**Executable specification:** A document describing requirements in a way that can be processed into automated tests.

# Feature: rendering executable specifications

## Example: simple Gherkin example

*Given* a project with a feature file
*When* I run the test suite
*And* I package the documentation
*Then* the package contains the lines from this feature file

# Feature: Rendering README

## Scenario: My scenario

*Given* a
*When* b
*Then* c