

# Doc Packager

# Context

# 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, step 3 is automated in GitHub Actions. This way, each commit leads to a new PDF explaining the project. This PDF is accessible in the [Package docs](#) workflow history and [in the repository's GitHub Pages](#).

It is currently an experiment, not useable in production. Contact [Sander](#) if you have questions or want to collaborate.

# 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.

# Requirements

## **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: PDF**

### **Example: Rendering BPMN**

*Given* a BPMN model

*When* I render it to PDF

*Then* I have a PDF file

### **Example: Writing PDF**

*Given* a project

*When* I execute the program

*Then* I get a PDF

# **Feature: Rendering README**

## **Scenario: My scenario**

*Given a*

*When b*

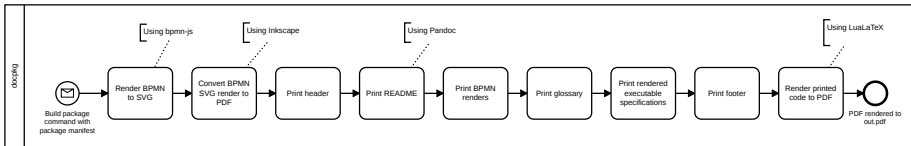
*Then c*



# Solution design

# Business processes

## Building a package



# Validation

# Implementation blueprint