Write great documents

Thomas Lutz

Write great documents:

Thomas Lutz

Table of Contents

List of Figures

Introduction

As a software engineer I love to work with source code that comes with Unit Tests and a well written documentation. Markdown is surely a good way to write your documents. But sometimes it comes in handy to have a shiny PDF to store offline, send via email or print in high resolution. For those cases Docbook (co-founder O'Reilly Media [https://www.oreilly.com/]) is a good solution because it supports different output formats like pdf, epub or html. My project livingfire-docbook [https://gitlab.com/phoen1x/livingfire-docbook] shows a easy way to get started with Docbook and alter this example [https://gitlab.com/phoen1x/livingfire-docbook/raw/master/book.pdf] for your own needs. The software can also covert Markdown to Docbook using pandoc [http://pandoc.org/] and combine multiple Markdown files like a GitLab Wiki into one PDF. In addtion to that you can write UML diagrams in PlantUML [http://plantuml.com/] and have them rendered into pictures in your book.

Prerequirements

Make sure you have a working Docker [https://docs.docker.com/engine/installation/] and docker-compose [https://docs.docker.com/compose/install/] environment.

Usage

```
# download project files
git clone https://gitlab.com/phoen1x/livingfire-docbook.git
cd livingfire-docbook

# start livingfire-docbook
docker-compose up -d

# convert docbook to pdf
docker-compose exec docbook /book/convertBook.sh
xdg-open book/target/docbkx/pdf/book.pdf

# write Markdown or import wiki
cd tmp/wiki
git init
git remote add origin https://gitlab.com/phoen1x/livingfire-docbook.wiki.git
git pull
rm -rf .git
cd ../..

# convert plantuml to images - http://plantuml.com/
docker-compose exec docbook /book/convertPlantuml.sh

# convert markdown to docbook
docker-compose exec docbook /book/convertWiki2Docbook.sh
xdg-open book/target/docbkx/pdf/book.pdf

# stop  livingfire-docbook and get PDF
docker-compose down
cat book/target/docbkx/pdf/book.pdf > book.pdf
```

Configuration

Directory structure

```
book
|-- pom.xml
`-- src
    `-- main
        `-- documentation
            `-- docbook
            |   |-- book.xml
            |   |-- template_chapter.xml
            |   `-- chapters
            |       |-- chapter_1.xml
            |       |-- chapter_2.xml
            |       |-- chapter_n.xml
            |       `-- media
            |           |-- cover.png
            |           `-- logo.png
            |-- fonts
            |-- xsd
            `-- xslt
                `-- docbook.xsl
```

Probably everything you ever need to write a book is within the livingfire-docbook/book/src/main/documentation/docbook [https://gitlab.com/phoen1x/livingfire-docbook/blob/master/book/src/main/documentation/docbook] directory. Here you can find the book.xml [https://gitlab.com/phoen1x/livingfire-docbook/blob/master/book/src/main/documentation/docbook/book.xml] and a templates for your own content. To tweak the style of your book alter docbook.xsl [https://gitlab.com/phoen1x/livingfire-docbook/blob/master/book/src/main/documentation/xslt/docbook.xsl] and pom.xml [https://gitlab.com/phoen1x/livingfire-docbook/blob/master/book/pom.xml].

Writing Docbook

Write your documents with Mardown in a GitHub Page [https://pages.github.com/], GitLab Wiki [https://about.gitlab.com/features/], Golumn Wiki [https://github.com/gollum/gollum/wiki], ... and turn them with DocBook [http://docbook.org/] into into a high quality PDF [https://gitlab.com/phoen1x/livingfire-docbook/raw/master/book.pdf] using these links:

- DocBook Cheat Sheet [https://workaround.org/docbook-reference/]
- DocBook Tutorial [http://www.vogella.com/tutorials/DocBook/article.html]
- Page layout [http://www.sagehill.net/docbookxsl/PrintOutput.html]
- Headers and Footers [http://www.sagehill.net/docbookxsl/PrintHeaders.html]
- Maven + Docbook [http://docbkx-tools.sourceforge.net/]

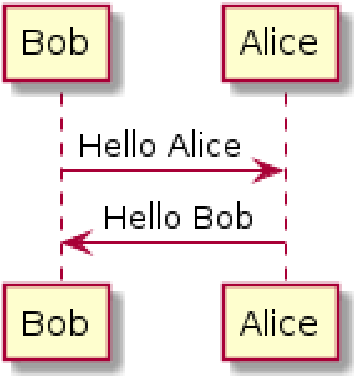
Plantuml

The project is also capable of converting PlantUML [http://plantuml.com/]

```
@startuml
Bob   ->  Alice : Hello Alice
Alice ->  Bob   : Hello Bob
@enduml
```

into an image

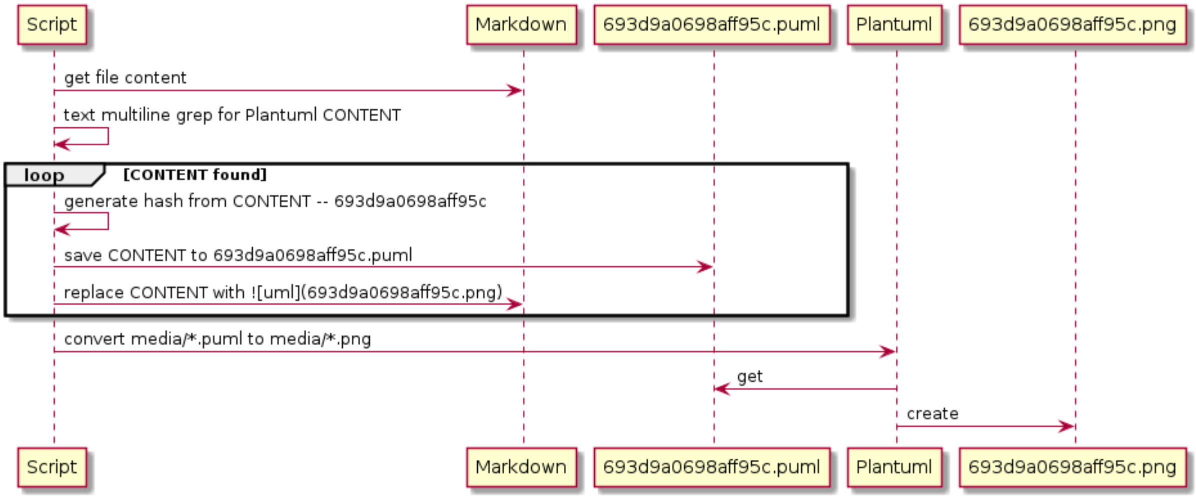
Figure 1. uml



example - Replace Markdown in file system

```
@startuml
Script -> Markdown: get file content
Script  -> Script : text multiline grep for Plantuml CONTENT
loop CONTENT found
    Script  -> Script : generate hash from CONTENT -- 693d9a0698aff95c
    Script  -> 693d9a0698aff95c.puml: save CONTENT to 693d9a0698aff95c.puml
    Script  -> Markdown: replace CONTENT with ![uml](693d9a0698aff95c.png)
end
Script  -> Plantuml: convert media/*.puml to media/*.png
Plantuml -> 693d9a0698aff95c.puml: get
Plantuml -> 693d9a0698aff95c.png: create
@enduml
```

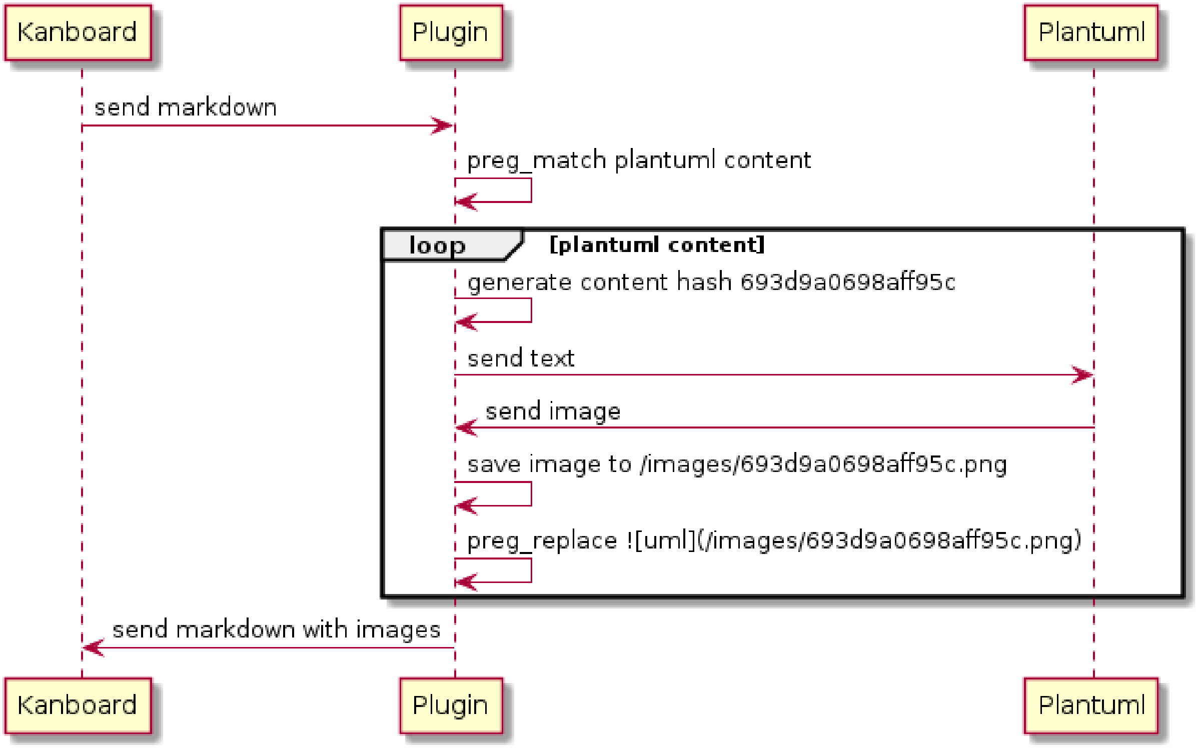
Figure 2. Sequence Diagram


Script
Markdown
693d9a0698aff95c.puml
Plantuml
693d9a0698aff95c.png
get file content
text multiline grep for Plantuml CONTENT
loop
[CONTENT found]
generate hash from CONTENT -- 693d9a0698aff95c
save CONTENT to 693d9a0698aff95c.puml
replace CONTENT with ![uml](693d9a0698aff95c.png)
convert media/*.puml to media/*.png
get
create
Script
Markdown
693d9a0698aff95c.puml
Plantuml
693d9a0698aff95c.png

example - Replace Markdown online

```
@startuml
Kanboard -> Plugin: send markdown
Plugin -> Plugin: preg_match plantuml content
    loop plantuml content
        Plugin -> Plugin: generate content hash 693d9a0698aff95c
        Plugin -> Plantuml: send text
        Plantuml -> Plugin: send image
        Plugin -> Plugin: save image to /images/693d9a0698aff95c.png
        Plugin -> Plugin: preg_replace ![uml](/images/693d9a0698aff95c.png)
    end
Plugin -> Kanboard: send markdown with images
@enduml
```

Figure 3. Sequence Diagram


Kanboard
Plugin
Plantuml
send markdown
preg_match plantuml content
loop
[plantuml content]
generate content hash 693d9a0698aff95c
send text
send image
save image to /images/693d9a0698aff95c.png
preg_replace ![uml](/images/693d9a0698aff95c.png)
send markdown with images
Kanboard
Plugin
Plantuml

More information on Plantuml can be found at the PlantUML website [http://plantuml.com/].

Converter

The source code of my converter can be found here [https://gitlab.com/phoen1x/livingfire-docbook/blob/master/book/src/main/java/de/livingfire/PlantumlConverter.java].

Gitlab Wiki / Markdown / Pandoc

To manually convert Markdown you need to install pandoc [http://pandoc.org/]

```
# apt-get install pandoc
pandoc --chapters -t docbook --output page.xml page.md
```

Usage without Docker

Maven

```
# install Java -- https://java.com/en/download/
git clone https://phoen1x@gitlab.com/phoen1x/livingfire-docbook.git
cd livingfire-docbook

# Linux
./mvnw clean site
xdg-open target/docbkx/pdf/book.pdf

# Windows
mvnw.cmd clean site
# Open target/docbkx/pdf/book.pdf in file browser
```

Note

This book was written in a GitLab Wiki [https://gitlab.com/phoen1x/livingfire-docbook/wikis/home]. Feel free to add your improvements via Merge Request [https://gitlab.com/phoen1x/livingfire-docbook] or open an issue [https://gitlab.com/phoen1x/livingfire-docbook/issues] if you find a bug.