riki static site generator

Subset of ikiwiki rewritten in Rust

2022-08-17 07:03

Contents

1 Introduction

`riki` is a small subset of ikiwiki[1] rewritten in Rust, for speed. This document describes the requirements and acceptance criteria for the software, and how to verify that riki meets them in an automated way. This is done using the Subplot[2] software.

2 Verification scenarios

The approach used for verifying acceptance criteria is to run `riki` against known inputs, and check that the output is as expected. Specifically this is done by comparing the Pandoc abstract syntax trees of the input and output. Pandoc is a well-known, well-respected tool that we rely on as an "oracle".

2.1 Markdown features

2.1.1 Empty Markdown file

Requirement: Given an empty input Markdown file, the output must be an empty HTML file.

given an installed riki
and file **site/empty.mdwn** from **empty**
when I run **riki build --plain-body site output**
then AST of **site/empty.mdwn** matches that of **output/empty.html**

File: **empty**

[1]

2.1.2 Plain text

Requirement: Given a Markdown file with plain text, the output must be an HTML file with the same text, without extra elements.

given an installed riki
and file **site/page.mdwn** from **para**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

[1] `http://ikiwiki.info/`
[2] `https://subplot.tech/`

File: **para**

```
Hello, world.

There are two paragraphs.
```

2.1.3 Quoted block

_Requirement: Given a Markdown file with an quoted block of text, the output must have a blockquote element.

given an installed riki
and file **site/page.mdwn** from **blockquote**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **blockquote**

```
> This is a quoted block.
```

2.1.4 Indented code block

_Requirement: Given a Markdown file with an indented code block, the output must have a pre element.

given an installed riki
and file **site/page.mdwn** from **indented-code**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **indented-code**

```
    This is indented by four spaces.
```

2.1.5 Fenced code block

_Requirement: Given a Markdown file with a fenced code block, the output must have a pre element.

given an installed riki
and file **site/page.mdwn** from **fenced-code**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **fenced-code**

```
```
This is a fenced code block.
```
```

2.1.6 Image link

_Requirement: Given a Markdown file linking to an image, the output must have an img element.

given an installed riki
and file **site/page.mdwn** from **image-link**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **image-link**

```
1 ![my kitten](cat.jpg)
```

2.1.7 Emphasised text

Requirement: Inline markup for emphasis must result in an em element in HTML output.

given an installed riki
and file **site/page.mdwn** from **emph**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **emph**

```
1 There is *emphasized*, and so is _this_.
```

2.1.8 Strongly emphasised text

Requirement: Inline markup for strong emphasis must result in a strong element in HTML output.

given an installed riki
and file **site/page.mdwn** from **strong**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **strong**

```
1 There is **emphasized**, and so is __this__.
```

2.1.9 Strike through in text

Requirement: Inline markup for strike through must result in a del element in HTML output.

given an installed riki
and file **site/page.mdwn** from **strike**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **strike**

```
There is ~~struck through~~.
```

2.1.10 Headings

Requirement: Given a Markdown file with headings of various levels, the output must be an HTML file with corresponding `h1`, `h2`, etc, elements, without extra elements. Up to six levels of headings must be supported.

given an installed riki
and file **site/page.mdwn** from **headings**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **headings**

```
# Heading one
## Heading two
### Heading three
#### Heading four
##### Heading five
###### Heading six
```

2.1.11 Inline code

Requirement: Inline code markup with backticks must result in a code element in HTML output.

given an installed riki
and file **site/page.mdwn** from **backticks**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **backticks**

```
There is `code` lurking here.
```

2.1.12 Table

Requirement: Markup of a table result in a table element in HTML output.

Note: This is disabled. Pandoc doesn't seem to handle the HTML table OK.*

```
given an installed riki
given file site/page.mdwn from table
when I run riki build --plain-body site output
then AST of site/page.mdwn matches that of output/page.html
```

2.1.13 Horizontal rule

Requirement: Markup of a horizontal rule must result in hr element in HTML output.

given an installed riki
and file **site/page.mdwn** from **rule**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **rule**

```
foo

---

bar
```

2.1.14 Unordered list

Requirement: Markup of an unordered list must result in a ul element in HTML output.

given an installed riki
and file **site/page.mdwn** from **ul**
when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **ul**

```
* first
* second
```

2.1.15 Ordered list

Requirement: Markup of an ordered list must result in an ol element in HTML output.

Note: This is disabled. Pandoc doesn't seem to parse the HTML list the same as the Markdown.*

```
given an installed riki
given file site/page.mdwn from ol
when I run riki build --plain-body site output
then AST of site/page.mdwn matches that of output/page.html
```

2.1.16 Task list

Requirement: Markup of a task list must result in a ul element in HTML output.

given an installed riki
and file **site/page.mdwn** from **tasklist**

when I run **riki build --plain-body site output**
then AST of **site/page.mdwn** matches that of **output/page.html**

File: **tasklist**

```
* [ ] not done
* [x] done
```

2.1.17 Definition list

Requirement: Markup indicating use of a definition list should be flagged as an error.

Justification: Neither the CommonMark specification, nor GitHub Flavored Markdown, supports definition lists, even though some Markdown variants do. The Markdown parser Riki uses doesn't support it.

given an installed riki
and file **site/page.mdwn** from **dl-1**
when I try to run **riki build --plain-body site output**
then command fails
and stderr contains "**definition list**"
given file **site/page.mdwn** from **dl-2**
when I try to run **riki build --plain-body site output**
then command fails
and stderr contains "**definition list**"
given file **site/page.mdwn** from **dl-3**
when I run **riki build --plain-body site output**
then file **output/page.html** contains "**: bar**"

File: **dl-1**

```
foo
: bar
```

File: **dl-2**

```
foo

: bar
```

File: **dl-3**

```
foo

<!-- no colon at beginning of line here -->: bar
```

2.1.18 Wiki links to other pages on the site

Requirement: Pages can link to other pages on the site, the same way ikiwiki does, including subpages.

given an installed riki
and file **site/dir/foo.mdwn** from **foo**
and file **site/absolute.mdwn** from **empty**
and file **site/dir/sibling.mdwn** from **empty**
and file **site/dir/foo/child.mdwn** from **empty**
and file **site/dir/foo/child/grandchild.mdwn** from **empty**
when I run **riki build --plain-body site output**
then file **output/dir/foo.html** contains "**href="../absolute"**"
and file **output/dir/foo.html** contains "**href="sibling"**"
and file **output/dir/foo.html** contains "**href="foo/child"**"
and file **output/dir/foo.html** contains "**href="foo/child/grandchild"**"

Note the uppercase link to the `child` page in the test page below.

File: **foo**

```
[[/absolute]]
[[sibling]]
[[child]]
[[child/grandchild]]
[[CHILD]]
```

2.1.19 Wiki links to pages that don't exist

Requirement: Linking to a page that doesn't exist is an error.

given an installed riki
and file **site/dir/foo.mdwn** from **badlink**
when I try to run **riki build --plain-body site output**
then command fails

File: **badlink**

```
[[missing]]
```

2.2 Directives

2.2.1 `img`

Requirement: the `img` directive embeds an image in the generated HTML page.

given an installed riki
and file **site/index.mdwn** from **img**
and file **site/img.jpg** from **jpeg**
when I run **riki build site output**
then file **output/index.html** contains "**<img src="img.jpg"**"

File: **img**

```
[[!img img.jpg]]
```

File: **jpeg**

```
This is a dummy JPEG image.
```

2.2.2 `meta title`

Requirement: the `meta title` directive sets page title.

given an installed riki
and file **site/index.mdwn** from **meta**
when I run **riki build site output**
then file **output/index.html** contains "**<title>Yo</title>**"

File: **meta**

```
[[!meta title=Yo]]
```

2.2.3 `shortcut`

Requirement: the `shortcut` directive created a shortcut that looks like a directive.

given an installed riki
and file **site/index.mdwn** from **shortcut**
when I run **riki build site output**
then file **output/index.html** contains "**foo!123**"

File: **shortcut**

```
[[!shortcut name="foo" url="https://example.com/foo/%s" desc="foo!%s"]]

[[!foo 123]]
```

2.3 Source file tree

2.3.1 Listing source files

Requirement: source files can be listed.

given an installed riki
and file **site/index.mdwn** from **empty**
and file **site/img.jpg** from **empty**
when I run **riki list site**
then stdout contains "**img.jpg**"
and stdout contains "**index.mdwn**"

2.3.2 Exclude unusual files

Requirement: files and directories that aren't meant to be part of the site content should be excluded.

given an installed riki
and file **site/index.mdwn** from **empty**

and file **site/img.jpg** from **empty**
and file **site/.git** from **empty**
and file **site/index.mdwn~** from **empty**
and file **site/#index.mdwn#** from **empty**
when I run **riki list site**
then stdout contains "**img.jpg**"
and stdout contains "**index.mdwn**"
and stdout doesn't contain "**.git**"
and stdout doesn't contain "**index.mdwn~**"
and stdout doesn't contain "**#index.mdwn#**"

2.4 Input files other than Markdown

Requirement: Input files that aren't Markdown files must be copied into the destination directory as-is.

given an installed riki
and file **site/image.jpg** from **image**
when I run **riki build --plain-body site output**
then files **site/image.jpg** and **output/image.jpg** match

File: **image**

```
# Dummy
Pretend this is an image.
```

2.5 Input files in sub-directories

Requirement: If an source page or file is in a sub-directory, it should be put in the corresponding sub-directory in the target directory.

given an installed riki
and file **site/foo/page.mdwn** from **image**
and file **site/bar/image.jpg** from **para**
when I run **riki build --plain-body site output**
then AST of **site/foo/page.mdwn** matches that of **output/foo/page.html**
and files **site/bar//image.jpg** and **output/bar/image.jpg** match

2.6 Output directory tree

2.6.1 No markdown files in output tree

Requirement: Markdown files are not copied to the output tree.

given an installed riki
and file **site/index.mdwn** from **empty**
when I run **riki build site output**
then file **output/index.html** exists
and file **output/index.mdwn** does not exist

2.6.2 Output files have source file modification times

Requirement: Files in the output directory have the same time stamp as the corresponding files in the source directory.

Note that due to limitations in the Subplot `lib/files` library, our check for modification times is imprecise.

given an installed riki
and file **site/index.mdwn** from **empty**
and file **site/index.mdwn** has modification time **1970-01-01 00:00:00**
and file **site/index.jpg** from **empty**
and file **site/index.jpg** has modification time **1970-01-01 00:00:00**
when I run **riki build site output**
then file **output/index.html** has a very old modification time
and file **output/index.jpg** has a very old modification time