

# mdbook-satysfi Documentation(-ja)

puripuri2100

概要

mdbook-satysfi のドキュメント

## 目次

1. mdbook-satysfi	2
2. 出力するファイルの名前を変更する	3
3. 読み込むパッケージを指定する	4
4. HTML タグを変換する	5
5. クラスファイルを自作する	7
6. SATySFi を呼び出す	10

# 1. mdbook-satysfi

## mdbook-satysfi

**mdbook-satysfi** は markdown ファイルから HTML ファイルで構成された *book* を生成する Rust 製のソフトウェアである **mdbook** の拡張機能を提供するソフトウェアです。

**mdbook-satysfi** をインストールした状態で `book.toml` に `[output.satysfi]` という記述を追加して `mdbook build` を実行すると、`book` の内容と同じ内容の SATySF*i* のドキュメントファイルが生成されます。

## 実装

実装は Rust で行っています。

リポジトリは `puripuri2100/mdbook-satysfi` です。

## 依存するもの

インストールには Rust と Cargo を必要とします。

また、生成されたドキュメントファイルを処理するためには `class-mdbook-satysfi` という SATySF*i* 用のライブラリの `v0.2.0` が必要であり、これのインストールには `satyrogaphos` が必要です。当然のことながら SATySF*i* も必要です。

## インストール方法

Rust と Cargo をインストールした状態で

```
cargo install mdbook-satysfi
```

と行うことで最新版が手に入ります。

## License

**mdbook-satysfi** は MIT ライセンスのもと公開されています。

## 2. 出力するファイルの名前を変更する

### 出力するファイルの名前を変更する

book.toml に以下のような内容を加えることで出力される SATySFi の文書ファイルの名前を変更することができます。

```
[output.satysfi]
output-file-name = "name.saty";
```

# 3. 読み込むパッケージを指定する

## 読み込むパッケージを指定する

book.toml に以下のような内容を加えることで読み込むパッケージを追加することができます。

`div` タグや `span` タグで独自のタグを作るときに、それに対応するコマンドを用意したいときに便利です。

```
[output.satysfi]
require-packages = ["base/int"]
import-packages = ["local"]
```

`require-packages` では `string array` を与えます。指定したパッケージが `@require:` で読み込まれます。

`import-packages` では、指定したパッケージが `@imports:` で読み込まれます。

# 4. HTML タグを変換する

## HTML タグを変換する

markdown ファイル内には HTML タグを直接書くことができます。

ここではその HTML タグを処理するための方法を説明します。

ただし、mdbook-satysfi は markdown ファイルを一旦 HTML に変換した後に処理をする方法を取っているため、意図しないタグの衝突が起こる可能性があることに充分注意して下さい。

book.toml 内に以下のようなコードを書いてください。

```
[output.satysfi]
[output.satysfi.html]
  [output.satysfi.html.hoge]
    command_name="fuga"
    children_type="inline"
    [[output.satysfi.html.hoge.attribute]]
      name = "src"
      type = "link option"
    [[output.satysfi.html.hoge.attribute]]
      name = "title"
      type = "inline"
```

`[output.satysfi.html.<tag-name>]` とすることで、そのタグが実際に書かれていた場合にその下を書いた設定が適用されます。

`div` タグと `span` タグの場合は `class` で指定した名前が使われます。

`command_name` はコマンドを出力する際にどうするかを設定します。デフォルトはタグ名です。

`children_type` は子要素がどうなるかを表しています。書かないと子要素は無視されます。与えられるのは

- `inline`
- `block`
- `inline list`
- `block list`
- `inline code`
- `block code`

だけです。

`[[output.satysfi.html.<tag-name>.attribute]]` とすることで属性を SATySFi コマンドの引数に変換することができます。

属性はこのテーブルに追加した順に渡されます。

`name` は属性の名前です。

`type` は属性の値がどうなるかを表しています。`* option` のとき、その属性がなかった場合は `None` になります。`option` が付いていないでその属性がなかった場合はその部分は無視されます。

与えられるのは

- `string`
- `link`
- `int`
- `bool`
- `inline`
- `block`

とそれぞれの `option` です。

`link` とは相対パスのことで、画像タグの処理などに使うことを想定しています。与えられた相対パスをその原稿の書かれた md ファイルのパスと結合して SATySFi ファイルに書き出します。

# 5. クラスファイルを自作する

## クラスファイルを自作する

mdbook-satysfi によって出力されたファイルを処理するためのクラスファイルを自作することができます。

### 必要なコマンドなど

```
(|  
  title : inline-text;  
  authors : inline-text list;  
  description : inline-text option;  
  language : string option;  
|)
```

というレコード型を一つ目の引数に、`block-text` を二つ目の引数に持ち、`document` 型を返す、`document` という名前の関数を提供する必要があります。

`language` フィールドに入力される値は `Some('ja')` や `Some('en')` といったものです。

実装する必要があるコマンドは以下の通りです。

- `+Chapter: [(int list) option; int; inline-text; block-text]` : 章を表すコマンドです。引数はそれぞれ「何番目の章の子や孫になっているのか」・「深さ」・「タイトル」・「中身」です。
- `+PartTitle: [inline-text]` : 目次に出力する部の分け目です。引数はタイトルです。
- `+Separator: []` : 目次に出力するラインを表します。引数はありません。
- `\strong: [inline-text]` : 強調を表します。
- `\emph: [inline-text]` : 強調を表します。
- `\strike: [inline-text]` : 打ち消し線を表します。
- `+block-quote: [block-text]` : 引用を表します。
- `\block-quote: [block-text]` : 引用を表します。
- `+heading: [int; inline-text]` : 節のタイトルを表します。一つ目の引数は深さです。
- `+p: [inline-text]` : 段落を表します。
- `\p: [inline-text]` : 段落を表します。
- `+code: [string]` : インラインコードを表します。
- `\code: [string]` : インラインコードを表します。

- `+code-block: [string option; string option; string]` : ブロックコードを表します。一つ目の引数は指定された言語を表し、二つ目の引数はカラーテーマを表します。ここに与えられる値は `book.toml` に `color-theme = "theme name"` と表記されると変更可能です。
- `\code-block: [string option; string option; string]` : 同上
- `\href: [string; inline-text]` : リンクを表します。
- `+img: [string; inline-text]` : 画像挿入を表します。二つ目の引数はキャプションです。
- `\img: [string; inline-text]` : 画像挿入を表します。二つ目の引数はキャプションです。
- `\footnote: [string]` : 脚注を出力します。引数は `key` です。
- `+add-footnote: [string; inline-text]` : 脚注の中身を登録します。一つ目の引数は `key` で、二つ目の引数が中身です。
- `+rule: []` : 線を出力します。
- `\task-list-marker: [bool]` : `true` のときはチェック印付きの四角を出力し、`false` のときは空の四角を出力します。`\item` コマンドの引数内に書かれます。
- `\item: [inline-text]` : 箇条書きの際に使用します。
- `\lisgint: [inline-text list]` : 箇条書きです。
- `+lisgint: [inline-text list]` : 箇条書きです。
- `\enumerate: [int; inline-text list]` : 数字付きの箇条書きです。一つ目の引数は「数字がどこから始まるか」を表します。
- `+enumerate: [int; inline-text list]` : 数字付きの箇条書きです。一つ目の引数は「数字がどこから始まるか」を表します。
- `+table: [block-text]` : 表を出力します。
- `+thead: [block-text]` : 表のヘッダーです。
- `+tbody: [block-text]` : 表の本体です。
- `+tr: [inline-text list]` : 表の横方向を指定します。
- `\th: [string option; inline-text]` : ヘッダーの中で表の一セルを表すコマンドです。一つ目の引数は位置を表します。中央そろえの時は `Some(`center`)` ・左揃えの時は `Some(`left`)` ・右揃えの時は `Some(`right`)` が与えられます。
- `\td: [string option; inline-text]` : `\th` のボディ版です。

## その他のコマンドについて

`mdbook-satysfi` はマークダウンを HTML に変換してから `SATySFi` コードに変換を



する形を取っています。そのため、原理的には HTML タグ全てに対応するコマンドが必要です。ですが、それらに対応するかどうかはクラスファイル作成者に任せられます。

また、`div` タグと `span` タグでは `class` で指定した名前がコマンド名として使われる仕様のため、それらが使用されているマークダウンファイルを処理したい場合はそれに対応するコマンドを実装する必要があるかもしれません。

## クラスファイルの変更の仕方

[`output.satysfi`] 以下に

```
class-file-name = "class-file-folder/class-file-name"  
is-class-file-require = true
```

のように記述します。

`class-file-name` にはクラスファイルの名前を文字列型で入力します。`class-file-name` フィールドが無い、もしくは文字列型ではない場合は `class-mdbook-satysfi/mdbook-satysfi` が使用されます。

`is-class-file-require` には「クラスファイルの読み込みが `@require` かどうか」を真偽値で入力します。`true` の場合は `@require` で読み込まれ、`false` の場合は `@import` で読み込まれます。デフォルト値は `true` です。

# 6. SATySF*i* を呼び出す

## SATySF*i* を呼び出す

[output.satysfi] テーブルの中に `pdf = true` というコードを入れると、自動で SATySF*i* が呼び出され、出力した `saty` ファイルがコンパイルされて PDF ファイルが生成されます。

デフォルトでは `pdf = false` で、SATySF*i* は呼び出されません。

## 高度な設定

```
[output.satysfi]
[output.satysfi.pdf]
# 各種設定
```

とすることで高度な設定を行うことができます。

## SATySF*i* の起動コマンドを変更する

以下のように `commands` という key に対して SATySF*i* を起動するためのコマンドを文字列か文字列のリストで与えることで SATySF*i* の起動コマンドを変更することができます。デフォルトは `satysfi` です。

```
[output.satysfi]
[output.satysfi.pdf]
commands = ["wsl", "satysfi"]
```

上のように、`["wsl", "satysfi"]` と与えると、SATySF*i* を起動するために

```
wsl satysfi main.saty
```

というコマンドが実行されます。

このままでは Linux でも `wsl` コマンドが呼ばれてしまうため、OS ごとに切り替えることができるようになっています。

```
[output.satysfi]
[output.satysfi.pdf]
commands = {
  windows = ["wsl", "satysfi"],
  macos = ["satysfi"],
  linux = "satysfi",
}
```

```
others = "satysfi"
}
```

のようすることで OS ごとに呼び出しコマンドを切り替えることができます（この場合も、コマンド名として文字列または文字列のリストを与えます）。

現在、選択できるのは

- Windows
- MacOS
- Linux

の 3 種類のみで、それ以外の OS は `others` でひとまとめにされています。

省略した場合はデフォルトの `satysfi` が起動コマンドとして選択されますので、

```
[output.satysfi]
[output.satysfi.pdf]
commands = {windows = ["wsl", "satysfi"]}
```

と書けば「Windows のときには `wsl satysfi <file name>` で起動させ、それ以外の OS では `satysfi <file name>` で起動させると設定できます。

## SATySFi に与えるオプションの変更

SATySFi に与えるオプションを変更することができます。

- 変更するために TOML に与える時に使うキーの名前
- 対応して変更される SATySFi のオプション
- キーに対応して与える値の種類
- デフォルトの値

の一覧のリストは以下のようになっています。

なお、`text-mode-configs` で文字列を与える場合は、

```
[output.satysfi]
[output.satysfi.pdf]
text-mode-cofigs = "tex,latex"
```

のようにカンマ区切りで文字列を与えてください。

文字列のリストで与える場合はそのまま

```
[output.satysfi]
[output.satysfi.pdf]
text-mode-cofigs = ["tex", "latex"]
```

のように与えてください。

また、`config-path` に与えるパスは「“`{build-dir}/satysfi`” フォルダ」からの相対パスにしてください。

`output-file-name` に与えるファイルのパスは `config-path` 同様に、「“`{build-dir}/satysfi`” フォルダ」からの相対パスにしてください。

キー	SATySFi のオプション	値	デフォルト
<code>is-bytecomp</code>	<code>--bytecomp</code>	真偽値	<code>false</code>
<code>is-type-check-only</code>	<code>--type-check-only</code>	真偽値	<code>false</code>
<code>is-debug-show-bbox</code>	<code>--debug-show-bbox</code>	真偽値	<code>false</code>
<code>is-debug-show-space</code>	<code>--debug-show-space</code>	真偽値	<code>false</code>
<code>is-debug-show-block</code>	<code>--debug-show-block</code>	真偽値	<code>false</code>
<code>bbox</code>	<code>bbox</code>		
<code>is-debug-show-block-space</code>	<code>--debug-show-block-space</code>	真偽値	<code>false</code>
<code>is-debug-show-overfull</code>	<code>--debug-show-overfull</code>	真偽値	<code>false</code>
<code>is-full-path</code>	<code>--full-path</code>	真偽値	<code>false</code>
<code>is-show-fonts</code>	<code>--show-fonts</code>	真偽値	<code>false</code>
<code>is-no-default-config</code>	<code>--no-default-config</code>	真偽値	<code>false</code>
<code>fig</code>	<code>fig</code>		
<code>config-path</code>	<code>--config</code>	文字列	オプションがつかない
<code>page-number-limit</code>	<code>--page-number-limit</code>	数値	オプションがつかない
<code>output-file-name</code>	<code>--output</code>	文字列	オプションがつかない
<code>text-mode-configs</code>	<code>--text-mode</code>	文字列・文字列のリスト	オプションがつかない