Title in command line option

目次

1 本書の目標

本書は、以下の事柄を実現できる方法を説明することを目標にします。

- 付番
  - 章や節・図・表・リスティングに自動的に付番する。
  - 参照する時、参照対象の番号を自動的に挿入する。
- 書式制御
  - 付番の書式を制御する。
  - 参照の書式を制御する。
  - パラグラフの書式を制御する。
- KDP で使える目次を作る。

- 図表一覧を作る。
  ただし、図表へのリンクがありません。用いるツールによる制限です。

本書で説明する方法により EPUB と HTML を作れます。
本書では EPUB ファイルを表示するために Kindle Previewer を使います。別途インストールしてください。

1.1 変換してみる

早速 Pandoc を使ってマークダウンで書かれたファイルを EPUB に変換してみましょう。

以下の二つのファイルを同じフォルダ内に作ってください。sample.md はマークダウンによる原稿ファイル、crossref.yaml は pandoc-crossref の制御ファイルです。いずれのファイルも、文字コードが UTF-8 でなければなりません。

リスト 1: sample.md

```
#最初の章{#sec:the1st}


「[@sec:the1st]」を参照する。
```

リスト 2: crossref.yaml

```
# 全体パラメータ
# 参照に参照先へのリンクを付ける。
linkReferences: true

# セクション用パラメータ
# true: 章番号を付ける
numberSections: true
# 章とみなされる見出しレベル。1 なら「#xxx」が章である。
chaptersDepth: 1
# 章番号からいくつの見出しレベルに付番するかを指定する。2 なら章と節に付番される。
sectionsDepth: 2
```

コマンドプロンプト（いわゆる DOS 窓）を起動し、これらのファイルがあるフォルダに移動してから、下記のコマンドラインを入力して実行してください。

```
pandoc sample.md  \
  -s  \
  --filter pandoc-crossref  \
  --metadata title="Title in command line option"  \
  --metadata crossrefYaml=crossref.yaml  \
  --epub-title-page=false  \
```

```
-o result.epub
```

あっという間に変換が終わります。でき上がった result.epub をダブルクリックすると、KindlePreviewer で表示されます。

Kindle Previewer の 表示

2　基本的な使い方

本章では、図を例にとり、付番や参照の書き方などの基本的な事柄を説明します。

2.1　図への付番と参照

2.1.1　図への付番

図に番号を与えるには、以下のように書きます。

```
![caption](imageResourceUrl "imageTitle"){#fig:id}
```

この書き方の前半部分である「`![caption](imageResourceUrl "imageTitle")`」は、Markdown の書き方そのものです。`imageTitle` を省略してかまいません。`caption` を省略すると、図タイトルが表示されなくなるだけではなく、参照もできなくなります。

付番にかかわるのはその後ろに付け足された `{#fig:id}` です。

- 前半部分とこの部分は、空白で区切られていてはなりません。
- `#fig:` は、画像に付番するよう指示する部分で、常にこの通りに書かなければなりません。
- `id` は、この画像を識別するための任意の一語とみなせる文字列であり、文書中でユニークでなければなりません。
- `id` は、日本語でもかまいません。

`{#fig:id}` が付けられていない場合、つまり、Markdown のイメージ貼り付けそのままの場合、番号が付けられず、参照もできません。

2.1.2　図の参照

図を参照するには、以下のように書きます。

```
[@fig:id]
```

`id` には、画像に付けられた id を指定してください。

2.1.3　例

例として monument.jpg という画像ファイルを文書に取り込んで付番し参照する例を掲げます。

リスト 3: sample.md

```
![図のキャプション](monument.jpg){#fig:monumentId}

図の参照：「[@fig:monumentId]」
```

Crossref.yaml は、前章のものをそのまま使ってください。コマンドラインも同じです。

この Markdown は、monument.jpg というファイルを取り込んで monumentId という ID を与え、その次の行でそれを参照しています。変換して下図のように表示されれば成功です。

ab 図 1: 画像への付番の例.a—図の表示,b—参照の表示

表示が 2 ページに分かれているのは、Kindle の仕様によります。

2.2 Pandoc 制御ファイル

ここまで使ってきた Pandoc のコマンドラインは、長くて書くのが大変です。バッチファイルなどのスクリプトを作る手もありますが、それより見てわかりやすい制御ファイルを作りましょう。

リスト 4 に制御ファイルの例を掲げます。
ここまで使ってきたコマンドラインのオプションを設定するものです。このファイルの文字コードも UTF-8 でなければなりません。

リスト 4: pandoc.yaml

```
#コマンドラインは、「pandoc -d pandoc.yaml」です。

input-files:  # 入力ファイルは、sample.md です。
    - sample.md
output-file: result.epub  # 出力ファイルは、result.epub です。
standalone: true  # 出力を単体で表示できるようにします。
filters:
    - pandoc-crossref  # pandoc-crossref を使います。
metadata:
    crossrefYaml: "Crossref.yaml"  # pandoc-crossref の制御ファイルです。
    title: "Title in command line option"  # ドキュメントタイトルです。
    number-sections: false  # Pandoc にセクション番号を付けさせません。
```

このファイルを利用したコマンドラインは、以下のようにとてもシンプルになります。

```
$ pandoc -d pandoc.yaml
```

-d オプションにより Pandoc に制御ファイル名を知らせています。
Pandoc は、このファイルの内容をコマンドラインオプションの代わりに取り込んで Markdown ファイルを変

換します。

コマンドラインオプションを指定することにより、制御ファイルによる指定をオーバーライドすることもできます。
例えば、制御ファイルで指定されている EPUB ファイルではなく HTML ファイルに変換させるには、以下のようなコマンドラインを使います。太字部分で指定した出力ファイル名が制御ファイルで指定されているものをオーバーライドして、`result.html` というファイルを出力します。

```
$ pandoc -d Pandoc.yaml -o result.html
```

これをブラウザで表示すると、図 2 のようになります。

HTML ファイルの方が EPUB より表示にかかる時間が短いので、執筆中には、変換結果をすばやく確認するために HTML ファイルを作るようお勧めします。
KDP で出版するならば、もちろん、最終的には EPUB を作っての確認が必要です。

3　様々な制御

前章では、pandoc-crossref により図に番号を付けたり、それを参照したりする方法を説明しましたが、色々と直したいところも出てきます。日本語なのですからキャプションに「Figure」ではなく「図」と付けたいですし、参照のリンクが「fig. 1」と数字の部分だけに置かれていては読者に気付いてもらえないかもしれません。pandoc-crossref は、制御ファイルを使ってそれらを変更できます。本章では、前章同様、図を例に採ってこのような制御について説明します。

3.1　キャプション書式の制御

まず、キャプションの書式の構造を説明します。
pandoc-crossref では、図のキャプション文字列の構造は、下図のようになっています。

図 3: 図キャプションの構造

「変数」が前置されているものは、pandoc-crossref の制御ファイルで設定できます。また、デフォルト値にある `$$xx$$` という表記は、「変数 `xx` の値」がその部分に挿入されることを表します。

表 1: 図キャプションの部品要素

役割	デフォルト	値
変数 `figureTemplate`	図キャプションの書式	`$$figureTitle$$ $$i$$$$titleDelim$$$$t$$`
変数 `figureTitle`	キャプションに前置される文字列	`Figure`
変数 `titleDelim`	前置される文字列と番号を区切る文字列	`:`
図の連番 i	図に付番された番号	`NA`

役割	デフォルト	値
キャプション t	Markdown 中に書かれたキャプション	NA

図 2 に表示されている図のキャプションは、変数 figureTemplate で指定される書式に従って作られた文字列です。デフォルト値では、変数 figureTitle が Figure 、変数 titleDelim が : ですから、図 2 に示したように、図に付けられるキャプションは「Figure 図番号:Markdown に書かれたキャプション」となります。ここで、「図番号」は、pandoc-crossref が自動的に図に与えた番号の文字列です。

図キャプションの書式は変数 figureTemplate で決まるのですから、そのデフォルト値がしているように変数 FigureTitle 等の他の変数を参照する必要は、必ずしもありません。例えば、変数 figureTemplate に含まれている $$titleDelim$$ を : に置き換えても、同じキャプションを作れます。変数 figureTemplate の値を書き換えるには、pandoc-crossref の制御ファイルに変数名をキーとするハッシュを書きます。前章のサンプルですと、下記の行を crossref.yaml に書き加えることになります。

リスト 5: 変数 titleDelim を使わない例

```
# このハッシュは、変数の値を書き換えます。
figureTemplate: $$i$$: $$t$$  # 変数 titleDelim の代わりに、直接「:」を書きました。
```

ただし、変数 titleDelim は表など他の種類のキャプションの区切り記号にも統一感を出すために使われていますので、できるだけこの変数を利用するようお勧めします。

3.1.1 番号の文字種

図番号の文字種を制御するには、変数 figLabels を用います。設定する値と番号の形式を下表に掲げます。

表 2: 番号文字の文字種

設定する値	制御ファイルでの書き方例	表示例	メモ
arabic	figLabel:arabic	1,2,3…	
roman	figLabel:roman	I,II,III,IV,…	
lowercase	figLabel:lowercase	i,ii,iii,iv,…	
alpha x	figLabel:alpha d	d,e,f,…	
列挙	figLabel:[イ，ロ，ハ，ニ]	イ, ロ, ハ, ニ	使うものすべてを列挙してください。

3.1.2 連番の振り方

図番号は、文書全体での連番、または、各章などの中での連番です。

変数 chapters により、どちらにするかを決められます。なお、変数 chapters は表や式などの付番にも影響しますので、ご注意ください。

表 3: 連番の振り方

変数 chapters の値	連番の振り方
false	文書内全体を通しての連番（デフォルト値）
true	章や節ごとに 1 番から始まる連番

変数 chapters が true の時には、以下のような形で連番が付けられます。

図 4: 章ごとの連番

章や節のどの範囲で連番を付けるかは、変数 chaptersDepth で決められます。この変数には、番号をリセットしたいヘッダレベルを設定してください。例えば、ヘッダレベル 1 を章に割り当てている場合、表 4 のようになります。

表 4: 連番の範囲

変数 chaptersDepth の値	連番の範囲
0	文書全体を通じて連番を付加する。
1	章ごとに番号をリセットし、図番号に章番号を付加する。デフォルト値。
2	節ごとに番号をリセットし、図番号に章番号と節番号を付加する。
3	以上項番号ごとなどになり、以下同様。

章ごとなどの連番の場合に章番号と連番を区切る文字は、変数 chapDelim の値で、デフォルト値は . です。

3.2 参照書式の制御

pandoc-crossref では、図の参照文字列の構造は、下図のようになっています。

図 2 に表示されている図の参照には、変数 refIndexTemplate に従って作られた文字列が使われています。デフォルト値では変数 figPrefixTemplate が $$p$$ $$i$$、変数 figPrefix が fig. ですから、「fig. 図番号」となります。なお、 は、改行を許さない空白文字です。

表 5: 参照文字列部品

要素	役割	デフォルト値
変数 refIndexTemplate	参照文字列の書式	$$i$$$$suf$$
変数 figPrefixTemplate	図番号の書式	$$p$$ $$i$$
変数 figPrefix	連番に付けるプレフィックス	fig., figs

要素	役割	デフォルト値
サフィックス suf	参照の中に書かれた suffix	NA
キャプション t	参照先の図のキャプション	NA
図の連番 i(figPrefixTemplate 内)	参照先の図に付けられた連番	NA
整形済連番 i(refIndexTemplate 内)	変数 figPrefixTemplate の評価値	NA

3.2.1 変数 figPrefix

参照に付けられるプレフィックスを指定する変数 figPrefix は、単数形と複数形を指定できるリストです。

複数形は、4.4 節で説明するように、複数の図を参照する場合に用いられます。制御ファイル内に書くと、以下のようになります。

```
figPrefix:
-fig.
-figs.
```

複数形がなければ、以下のように単なる変数としても問題ありません。

```
figPrefix: 図
```

3.2.2 サフィックス

サフィックスは、それぞれの参照で指定する、図番号の後ろに付ける文字列です。

通常の参照の書き方は [@fig:id] ですが、[@fig:idsuffix] のようにサフィックスとして任意の文字列を付け加えることができます。サフィックスをダブルクォーテーションで括らないでください。括ると、ダブルクォーテーションもサフィックスの一部とみなされます。サフィックス文字列の中に 2 個以上連続した空白文字がある場合、1 個にまとめられます。

例えば、図 2 にある参照を書く時に [@fig:monumentIdThisissuffix] と書けば、参照の文字列が以下のようになります。

```
fig. 1 This is suffix
```

3.2.3 プレフィックス変更

変数 figPrefix で決まる参照のプレフィックス部分を、それぞれの参照でオーバーライドできます。
通常プレフィックスには変数 figPrefix の値が使われて「fig. 1」となりますが、参照時に [myprefix@fig:monumentId] と書くと、変数 figPrefix の代わりに参照に書かれたプレフィックスが使われて myprefix1 となります。プレフィックスもサフィックスと同様に任意の文字列を書けます。ダ

ブルクォーテーションも文字列の一部とみなされることと複数の連続する空白が一つにまとめられることも同じです。

プレフィックスとサフィックスの両方を一つの参照に指定できます。

3.2.4 プレフィックス抑制

以下の書き方をすると、参照内にプレフィックスを表示しません。

リスト 6: プレフィックスを抑制する記法

```
[-@fig:monumentId]
```

このように書くと、図 5 の中段にある変数 `figPrefixTemplate` が省略されて、図番号が上段にある変数 `refIndexTemplate` の「`$$i$$`」の値として使われます。

「`-`」の後ろに空白文字を置くと、プレフィックス変更と解釈されてしまいます。間を開けずに書きましょう。

3.2.5 キャプションを含める

変数 `t` を用いて、参照書式に図のキャプションを含めることができます。変数 `t` は、変数 `refIndexTemplate` など書式を決める変数の定義の中でのみ使えます。

変数 `refIndexTemplate` のデフォルト値は「`$$i$$$suf$$`」ですが、これに「`$$t$$`」を付け加えて「`$$i$$$suf$$$$t$$`」とすると、参照書式が以下のようになります。

「`fig. 1` 図のキャプション」

3.3 種類ごとの参照書式

参照書式を決める変数 refIndexTemplate はシンプルな文字列として書式を保持していますから、それを変更すると、図だけではなく表や式などにも影響します。これを避けるには、この変数を、図専用の書式とそれ以外の書式を区別して保持するディクショナリーにします。

リスト 7:refIndexTemplate をディクショナリーにする

```
refIndexTemplate:
  fig: $$i$$$suf$$ $$t$$
  default: $$i$$$suf$$
```

このようにしておけば、図の参照書式だけにキャプションが表示されるようになります。

表や式などにもそれぞれ異なる参照書式を使いたい場合には、下表のように書いてディクショナリーに追加すれば実現できます。

変数 refIndexTemplate の値の中だけで使える変数 t・suf・i は、図と同様に表や式などでも使えます。

表 6: 種類ごとの参照書式

要素	書き方の例
図	`fig: $$i$$$$suf$$ $t$$`
表	`tbl: $$i$$:$$t$$`
式	`eq: $$i$$`
リスティング	`lst:$$i$$ $$t$$`
章や節	`suf: $$i$$$$suf$$$$t$$`
定義されていない要素すべて	`default: $$i$$$$suf$$`

3.4 参照の制御

この節では、書式以外の点で図などへの参照を制御する方法を説明します。

3.4.1 参照からリンクを張る

pandoc-crossref のデフォルトでは、図番号への参照から図へのリンクが張られません。リンクを張るには、変数 `linkReference` を `true` にしなければなりません。pandoc-crossref の制御ファイルである crossref.yaml に、以下の行を追加しましょう。

リスト 8: 参照からリンクを張るための設定

```
linkReference: true
```

この設定は、表や式などにも有効です。

3.4.2 プレフィックスもリンクに含める

参照のリンクは、図 2 に示すように、参照文字列の番号の部分だけにしか置かれていません。これではクリックしにくいですし、図番号と言えばプレフィックスまで含めて考えるのが普通でしょう。

変数 `nameInLink` を `true` にすることにより、プレフィックスにもリンクが置かれるようになります。pandoc-crossref の制御ファイルである crossref.yaml に、以下の行を追加しましょう。

リスト 9: 参照のプレフィックスにもリンクを置くための設定

```
nameInLink: true
```

このようにすると、図 2 の参照とは異なり図 6 のようにプレフィックスの部分にもアンダーラインが引かれて、リンクが置かれていることが分かります。

図 6: プレフィックスにもリンク

参照にサフィックスを書いている場合には、サフィックスの部分にもリンクが置かれます。

この設定は、表や式などにも有効です。

3.4.3 複数参照

複数の図に対する参照をまとめて書くことができます。

単独の参照: `[@fig:id1]`
複数の参照: `[@fig:id1;@fig:id2;…]`

参照を括る大括弧の中に複数の参照先をセミコロンで区切って書くだけです。

表 2 に示す番号の文字種が arabic の場合、連番の図が参照される時には全部を列挙するのではなく、それらをまとめて範囲を表示にされます。連番でなければ、カンマで区切って列挙されます。

例を下表に示します。

複数参照の例{#tbl: 複数参照の例}

参照する数	参照表記	参照文字列
1 個	`[@fig:id1]`	fig. 1
連番 2 個	`[@fig:id1;@fig:id2]`	figs.1,2
連番 3 個	`[@fig:id1;@fig:id2;@fig:id3]`	figs.1-3
飛び番 2 個	`[@fig:id1;@fig:id3]`	figs.1,3
飛び番 3 個	`[@fig:id1;@fig:id3;@fig:id5]`	figs.1,3,5
混合 4 個	`[@fig:id1;@fig:id2;@fig:id3;@fig:id5]`	figs.1-3,5

連番や飛び番の区切り文字は、以下の変数により設定できます。

連番や飛び番の区切り文字{#tbl: 連番や飛び番の区切り文字}

変数名	役割	デフォルト値	例	変更例	変更した表示
`rangeDelim`	3 個以上の連番の時に範囲を示す区切り文字	`-`	figs.1-3	`to`	figs.1to3
`pairDelim`	2 個を列挙する時の区切り文字	`,`	figs.1,3	`and`	figs.1 and 3
`refDelim`	3 個以上を列挙する時の最後以外の区切り文字	`,`	figs.1,3,5	`,`	figs.1,3,5
`lastDelim`	3 個以上を列挙する時の最後の区切り文字	`,`	figs.1,3,5	`, and`	figs.1,3, and5

それぞれの区切り文字と図番号の間には、空白文字を含めて自動的には何も表示されません。読みやすくするために、区切り文字の前後に適切な空白文字を含めるようお勧めします。変数 refDelim の変更例には、空白文字を置かなかった場合の表示を掲げてみました。

上記の変更例を設定する pandoc-crossref 設定ファイルの記述は、以下の通りです。

リスト 10: 連番などの区切り文字を設定する

```
rangeDelim: "to"
pairDelim: "and"
refDelim: ","
lastDelim:",and"
```

3.5 生成される HTML

本書では Pandoc が生成する HTML を掲載します。図や表に対して生成される HTML の構造が分かれば、CSS を使って望みの見た目を作れるからです。

本書で掲載する HTML は、Pandoc が生成したものそのままではなく、わかりやすいように整形しています。HTML タグの順序・入れ子・兄弟関係・属性などは生成されたままですが、改行を挿入したり、空白を挿入削除したりしています。

3.5.1 図番号の HTML

既出の通り、図番号付きで画像を取り込む Markdown は、以下の通りです。

リスト 11: 図番号付きの画像

```
![図のキャプション](monument.jpg "imageTitle"){#fig:monumentId}
```

この Markdown に対して、以下の HTML が生成されます。

リスト 12: 図番号付き画像の HTML

```
<figure id="fig:monumentId">
    <img src="monument.jpg" title="imageTitle" alt="図のキャプション">
    <figcaption>
        Figure1: 図のキャプション
    </figcaption>
</figure>
```

Markdown で指定した ID が figure タグの id 属性で設定されています。キャプションは、figcaption 要素と img 要素の alt 属性に設定されています。図を表示する img 要素とキャプションは、figure 要素内にまとめられています。第 4.5.2 項図番号参照の HTML 図を参照する Markdown は、以下の通りです。

リスト 13: 図参照

```
[@fig:monumentId]
```

この Markdown に対して生成される HTML は、図に番号がついている場合、以下の通りです。

リスト 14: 図参照の HTML

```
<a href="#fig:monumentId">fig. 1</a>
```

図の参照は、単純に a 要素で表現されています。

a 要素のコンテントは、4.2 節に示した方法で作られた文字列です。

3.5.2 表現を変更する

HTML の構造が分かれば、CSS で表現を変えることができます。例として、図 2 では画面の左側に寄っている図を中央寄せして、図の範囲が分かりやすいように線で囲ってみましょう。

以下の内容のファイル pandoc.css を作ってください。文字コードは、UTF-8 でなければなりません。

リスト 15:pandoc.css

```
<style>
    figure{
        padding-top: 0.5em;
        border: 1px solid #404040;
        text-align: center;
    }
</style>
```

そして、pandoc.yaml に以下の二行を追加します。

リスト 16:CSS 取り込み

```
include-in-header:
    - InHeader.css
```

`include-in-header` により、pandoc.css の内容が生成された EPUB や HTML に挿入されます。挿入場所は Pandoc が自動的に挿入する style 要素の直後ですから、元々 Pandoc が定義している表現をオーバーライドできます。

EPUB や HTML を生成させると、図 2 の表示が下図のように変わります。

図 2 と比べると、画像の表現が変わっていることがはっきりわかります。このように、生成される HTML の特徴を利用して表現を変えることができます。

なお、参照側は単純に a タグを使っていますので、CSS で表現を変えるとすべてのリンクの書式が変わってしまいます。図のリンクの書式だけを変えることはできません。

4　KDP 用原稿のための配慮

4.1　タイトルページを消す

KDP では、原稿登録時に、表紙を本文とは別に指定します。ですから、中表紙とみなすなら別ですが、EPUB に表紙を含めてはいけません。

Pandoc のデフォルトでは表紙を作るようになっていますから、これを消しましょう。

コマンドラインに `--epub-title-page=false` というオプションを追加して消します。

```
pandoc -d Pandoc.yaml --epub-title-page=false
```

Pandoc のドキュメントでは、このオプションを pandoc.yaml にも書けることになっていますが、残念ながら Version.3.1.2 で確認してみるとエラーになってしまいます。コマンドラインで指定しましょう。

4.2　目次を消す

KDP に EPUB を登録する時には目次が必須ですが、ナビゲーション用の目次が用意されていれば良いのであって、本文中に目次が含まれていなければならないということではありません。Pandoc で EPUB を作ると、ナビゲーション用の目次は必ず作られます。本文の一部として目次を作るか否かは著者が決めて問題ありません。特に書籍が大きい場合には目次が長くなりがちです。そのような時には目次を本文から削除したくなります。

目次を本文から削除するには、pandoc.yaml に以下の行を追加してください。

```
toc:false
```

これで本文から目次が削除されます。この設定にかかわらず、ナビゲーション用の目次は作られますので、KDP の要件は満たされます。

4.3　図表一覧を作る

Markdown の適当なところにコマンドを挿入することにより、図・表・リスティングの一覧を作れます。表とリスティングの一覧の作り方は図一覧とほぼ同一ですから、以下では図一覧について説明します。

表とリスティングの一覧については、表 25 に掲げる対応表をご覧ください。残念なことに、図表一覧から各図などへのリンクは、張られません。

4.3.1 図一覧の Markdown

図一覧の Markdown は、以下の通りです。

```
\listoffigures
```

5.7.3 項のリスト 38 で例示に使った図の一覧を作ると、図 44 の様に表示されます。

図 44: 図一覧

このように、図一覧には、図だけではなくサブフィギュアも含まれます。

4.3.2 図一覧の書式制御

4.3.2.1 図一覧のタイトル

図表一覧のタイトルは、変数 `lofTitle` の値です。
デフォルト値は `# List of Figures` です。
先頭の `#` は、レベル 1 のヘッダを表す Markdown 記法です。

4.3.2.2 図一覧本体の書式

図一覧の各行は、以下の構造により作られます。

図 45: 図一覧本体の書式

図番号は、図 5 に掲げる変数 `refIndexTemplate` による書式制御に従います。つまり、4.2 節で説明した図を参照する時と同じ書式になります。ただし、pandoc-crossref がプレフィックスとサフィックスを付けないので、結果的には図 45 の構造になっています。この構造は、プレフィックスを省略する参照の Markdown「`[-@fig:id]`」を書いた時と同じものです。

サブフィギュアの場合には、以下の構造により作られます。

図 46: 図一覧本体中サブフィギュアの書式

サブフィギュアがある場合には、図 40 に掲げる変数 `subfigureRefIndexTemplate` に従って作られた文字列が使われます。こちらも pandoc-crossref がプレフィックスとサフィックスを付けないので、結果的に図 46 の構造になっています。こちらも、プレフィックスを省略する参照の Markdown「`[-@fig:subfigid]`」を書いた時と同じものです。

表 24: 図一覧の各行の文字列の部品

要素	役割	デフォルト値
変数 `lofItemTitle`	各行の先頭に付ける文字列	空文字列
変数 `listItemTitleDelim`	参照文字列とキャプションを区切る文字列	.

他の部品については、5.2 節の表 9 と 5.7 節の表 22 をご覧ください。

4.3.3 生成される図一覧の HTML

図の一覧は、一つずつの図などごとに br 要素で改行されているだけの単なる文字列を div 要素で括ったものです。

リスト 42: 図一覧の HTML

```
<h1>ListofFigures</h1>
<div class="listlist-of-fig">
1. GroupCaption<br>

1 (a). BigCaption1<br>

1 (b). MiddleCaption1<br>

1 (c). SmallCaption1<br>

1 (d). BigCaption1<br>

</div>
```

図一覧のタイトルは、変数 `lofTitle` のデフォルト値の通り、レベル 1 のヘッダになっています。図一覧本体を括る div 要素には、図一覧であることを表すクラスが付けられます。

4.3.4 図表一覧の書式制御まとめ

表とリスティングの一覧も図一覧と同じ方法で作れます。表 25 に記法や書式制御変数の対応表を掲げます。

表 25: 図表一覧の記法対応表

一覧	Markdown 記法	行タイトル	番号の書式	div 要素のクラス
図	`\listoffigures`	変数 `lofItemTitle`	変数 `refIndexTemplate`	`list list-of-fig`
図（サブフィギュアの場合）	-	同上	変数 `subfigureRefIndexTemplate`	-
表	`\listoftables`	変数 `lotItemTitle`	変数 `refIndexTemplate`	`list list-of-tbl`
リスティング	`\listoflistings`	変数 `lolItemTitle`	変数 `refIndexTemplate`	`list list-of-lst`

変数 refIndexTemplate が連番の書式として図・表・リスティングのすべてに使われていますが、種類ごとに連番の書式を変えたい場合には 4.3 節で説明したように、変数 refIndexTemplate をディクショナリーにしてください。

5　付録

参考のために、本書で利用した制御ファイルと、それらを利用する際の注意事項を掲載します。

5.1　Pandoc 制御ファイル

リスト 43:Pandoc 制御ファイル

```yaml
input-files:
    - Prologue.md
    - Chapter1.md
    - Chapter2.md
    - Chapter3.md
    - Chapter4.md
    - Chapter5.md
    - Chapter6.md
    - Appendix.md
standalone: true
from: markdown+multiline_tables
filters:
    - pandoc-crossref
metadata:
    crossrefYaml: "crossref_ config. yaml"
    title: "Title in command line option"
    toc-title: "目次"
    language: "ja-JP"

include-in-header:
    - InHeader.css

toc: true
html-math-method:
    method: mathml

number-sections: false
variables:
```

```
    indent: true

toc: false
```

5.2 pandoc-crossref 制御ファイル

リスト 44: pandoc-corssref 制御ファイル

```
# セクション用パラメータ
numberSections: true
chaptersDepth: 1
sectionsDepth: 3

secPrefixTemplate: "$$i$$"
secLabels: arabic
secHeaderTemplate: "第$$i$$$$secHeaderDelim[n]$$$$t$$"
secHeaderDelim:
  - "章&#32;"
  - "節&#32;"
  - "項&#32;"
  - " "

# 図用パラメータ

figureTitle: "図 "
figPrefix: "図"
figPrefixTemplate: $$p$$$$i$$

# 表用パラメータ

tableTitle: "表"

tblPrefix: "表"
tblPrefixTemplate: $$p$$$$i$$

# 参照用パラメータ

refIndexTemplate: $$i$$$$suf$$

# リスティング用パラメータ
```

```
codeBlockCaptions: true

listingTitle: "リスト"
lstPrefix: "リスト"

# 全体用パラメータ

linkReferences: true
nameInLink: true
```

5.3 追加の CSS ファイル

本文中に掲載した HTML の構造を利用して、スタイルを適用する対象を選ぶようにしています。

リスト 45: 追加の CSS ファイル

```
<style>
    /*==== 全体調整 ====*/
    /* 全体の幅などを調整する */
    body {
        width: initial;
        max-width: 100%
        max-height: 100%
    }
    /* 改ページ */
    .breakpage {
        break-before: page;
    }

    p {
        margin: 0.7em 0;
        text-align: left;
    }

    /*==== ヘッダ ====*/
    h1 {
        break-before: page;
        margin: 0;
        border-bottom: 2px solid;
        break-after: avoid-page;
```

```
}
h2 {
    font-size: 1.6em;
    text-decoration: underline;
    break-after:
    avoid-page;
}
h3 {
    font-size: 1.5em;
    text-decoration: underline;
    break-after: avoid-page;
}
h4 {
    font-size: 1.4em;
    text-decoration: underline;
    break-after: avoid-page;
}
h5 {
    margin-block-end: 0em;
    font-size: 1.2em;
    font-weight: bold;
    break-after: avoid-page;
}

/*==== div 要素による制御 ====*/
/* div 要素によりパラグラフの位置と背景色を制御する。 */
div.indentedColoredBox {
    background: #f0f0f0;
    margin-left: 2em;
}

div.indented {
    margin-left: 2em;
}

/*==== 図の書式 ====*/

figure {
    text-align: center
```

```
}

/*==== リスティングの書式 ====*/
/* リスティングを少し右寄せにする。*/
div.listing {
    margin-left: 2em;
    padding: 0
}

/* キャプションをリスティング部分にぴったり寄せる。*/
div.listing > p {
    margin: 1em 0 0 0
}

/* リスティング部分の余分な余白をなくし、背景に色を付ける。*/
div.listing > pre {
    background: #f0f0f0;
    margin: 0
}

/*==== 図の書式 ====*/
figure > img {
    break-after: avoid-page;
}
img {
    border: 1px solid #404040;
}

/*==== リスト ====*/
/* リスト内のイメージを少し右に寄せる。*/
li > img {
    margin-left: 2em;
}

/*==== 表の書式 ====*/
table {
    margin: 0.5em 0;
    border-collapse: collapse;
    overflow-x: auto;
```

```
    display: block;
    border: 0 0;
    break-inside:
    avoid-page;
}
table caption {
    margin-bottom: 0;
    border-bottom: thick;
    font-weight: bold;
    break-after: avoid-page;
}
thead tr {
    border: 1px solid;
}
thead th {
    border-left: 1px dotted;
    border-right: 1px dotted;
    border-bottom: 2px solid;
}
thead + tbody {
    margin-top: initial;
    border-top: 2px solid #1a1a1a;
    border-bottom: 1px solid #1a1a1a;
}
thead + tbody tr {
    border: 1px solid;
}
thead + tbody td {
    border: 1px dotted;
}

/*==== リスティング ====*/
div.listing p {
    break-after: avoid-page;
}
code {
    font-size: normal;
}
```

```
    /*==== 汎用 ====*/

    /*==== 印刷 時 ====*/
    @media print {
        figure {
            break-inside: avoid;
        }
        div > table {
            break-inside: avoid-page;
        }
        div > table > tbody td {
            break-inside: avoid;
        }
        div {
            break-inside: avoid;
        }
    }
</style>
```

5.4 サンプル制御ファイル利用上のポイント

5.4.1 章番号などの参照

章番号などを参照する時に、単に [@sec:id] と書くと、参照文字列が「第 1」となってしまいます。本来は、「第 1 章」のように、末尾に「章」や「節」を表示すべきです。

pandoc-crossref には、自動的に末尾に「章」などを付ける機能ために使えるがありません。仕方がありませんので、参照する箇所の suffix として「章」などを付けています。

```
[@sec:id 章]
```

5.4.2 例示のコード

すぐ上に掲載したようなコードの例示は、HTML の機能を使って実現しています。マークダウン原稿では、以下のように書いています。

```
<div class="indentedColoredBox" >
    \[\@sec:id 章\]
</div>
```