SLγDIFl ドキュメント

monaqa
GitHub: https://github.com/monaqa

2020 年 7 月 23 日

はじめに

SLYDIFI とは

SLYDIFI は，SATYSFI でプレゼンテーション用のスライドを作成するためのツール．以下のような機能を備えている．

- タイトルフレーム，セクションフレームなどを含めたスライドの作成
- スライド内でのインラインマークアップ
- 任意の位置への図版の挿入，整列
- オーバーレイ
- カスタマイズ可能なテーマ

このドキュメントも SLYDIFI を用いて書かれているが，ドキュメント用にフォントサイズを縮めたスライドテーマを用いている．

SLyDIF$_{\mathrm{I}}$ の概要

パッケージ構成

SLγDIFI は以下のヘッダファイルで構成されている.

- `slydifi.satyh`: 共通処理やコマンドが書かれた `SlydifiScheme` モジュールを定義.
- `functions` ディレクトリ：すべてのテーマに共通の処理を機能ごとに分割して格納. ここにあるものは基本的に `slydifi.satyh` ですべて import される.
  - `color.satyh`: 色の操作に関するモジュール `SlydifiColor` を定義.
  - `figbox.satyh`: 図版に関するモジュール `FigBox` を定義.
  - `footnote.satyh`: 脚注に関するモジュール `FootnoteScheme` を定義.
  - `overlay.satyh`: オーバーレイに関するモジュール `SlydifiOverlay` を定義.
  - `param.satyh`: 汎用パラメータに関するモジュール `Param` を定義.
- `theme` ディレクトリ： SLγDIFI のテーマを格納. 内部で `slydifi.satyh` を読み込む.
  - `akasaka.satyh`
  - `hakodate.satyh`
  - `plain.satyh`

パッケージのインポート

基本的には，theme 内に入っているテーマファイルを読み込めばよい．

```
@require: class-slydifi/theme/plain
% "plain"の部分を変えてテーマ変更が可能

SlydifiPlain.document(|  % テーマを変えるにはモジュール名の変更も必要
|) '<
   (本文)
>
```

テーマはデフォルトで以下の三種類が用意されている．

- plain (module name: SlydifiPlain): 最も簡素なテーマ．
- hakodate (module name: SlydifiHakodate): Gruvbox の色をベースにしたテーマ．
- akasaka (module name: SlydifiAkasaka): 灰色のスタンダードなテーマ．

SLyDIFj の基本機能

フレーム：スライドの 1 ページ

フレームの作成には +frame{ title }< inner > を用いる.

```
+frame{フレーム：スライドの 1 ページ}<
  +p{
    フレームの作成には `+frame` を用いる.
  }
  ...
>
```

- +frame で囲った部分が出力される PDF の 1 ページになる.
- +frame の中にどれだけ多くの記述を入れても，自動でのページ分割は行われない.
  - 入り切らないブロックは描画領域からはみ出て見えなくなるだけ.
  - 「1 枚のフレームでどの情報まで表示するか」という発表時の戦略に関わる部分は手動で制御したほうが良い，という考えから.

章組み：タイトルフレーム

タイトルフレームを挿入したい箇所に `+make-title(| configs |)` を入れる．

```
+make-title(|
  title = {|\SLyDIFi; ドキュメント |};
  author = {
    | monaqa
    | GitHub: \link(`https://github.com/monaqa`);
    |};
  date = {|2020 年 7 月 23 日 |};
|);
```

- 引数は `title`，`author`，`date` ラベルを持つレコード型であり， 対応する値はいずれも `inline-text list` 型．
- `|` で区切ることで，複数行にわたるタイトルや著者名を表示可能

章組み：セクションフレーム

+section{| title |}< frames > を使うと，冒頭にセクションフレームが追加される．

```
+section{|\SLyDIFi; の基本機能 |}<

  +frame{フレーム：スライドの 1 ページ}<（内容）>
  +frame{章組み：タイトルフレーム}<（内容）>
  ...

>
```

- 現時点では，セクション冒頭にセクション用のフレームが追加されるだけ．
- 将来的には以下のような機能を追加するかもしれない．
  - スライド冒頭の目次に反映させる（現在は自動目次機能そのものがない）
  - 各スライドのヘッダーやフッターにセクション名を表示する

フレーム内の文書構造

- +p{ inner text } で段落を作成できる.
- +listing{ * items } で番号のない箇条書きを作成できる（enumitem パッケージを使用）.
- +enumerate{ * items } で番号付き箇条書きを作成できる（同上）.

```
+frame{パッケージ構成}<
  +p{
    \SLyDIFi; は以下のヘッダファイルで構成されている.
  }
  +listing{
    * `slydifi.satyh`: 共通処理やコマンドが書かれた
      `SlydifiScheme` モジュールを定義.
    * ...
  }
>
```

インラインマークアップ

SLyDIFi ではいくつかのインラインマークアップを標準で提供.

コマンド	役割
`\br;`	強制改行
`\emph{ text }`	強調 （強調の方式はテーマにより異なる）
`\text-color(color){ text }`	テキストの色の変更
`\uline{ text }`	テキストに下線を引く
`\strike{ text }`	~~打ち消し線~~
`\ctx( ctx-func ){ text }`	テキスト処理文脈の部分的な変更.
`\SLyDIFi;`	「SLyDIFi」 と書きたいときに

スライドのテーマによっては, `\textbf` など上で述べた以外のコマンドを追加していることもある. また, `annot` や `code` ライブラリで追加されるコマンドを用いることも可能.

オーバーレイ

オーバーレイ

1 枚のフレームの中で，プレゼンターの操作によって初めはなかった文字が現れたり，文字が順々に表示されたりすることがある．このようなフレームの動きをオーバーレイと呼ぶ．

最も単純なオーバーレイの実現方法としては，右のコードのように一部が共通しているフレームを複数作成することが挙げられる．しかしこの方法には以下の問題がある：

- 同じ内容を 2 回書かなければならない
- 共通する部分を変更する際に複数箇所を編集する必要がある

```
+frame{タイトル}<
  +p{1 枚目と 2 枚目で表示される段落．}
>
+frame{タイトル}<
  +p{1 枚目と 2 枚目で表示される段落．}
  +p{2 枚目のみで表示される処理．}
>
```

オーバーレイ

SLYDIFI では，右のように書くことでオーバーレイを実現できる．

+multiframe を使うと，第 1 引数に指定した数だけのレイヤーが作られる．第 2 引数のインラインテキストにはフレームタイトルを，ブロックテキストには +frame と同じように本文を指定する．複数枚のレイヤーを作成することで，「同じ内容のスライドを，一部だけ変えて繰り返す」ことが可能となる．

2 つ目の +p ではオプション引数が指定されており，オプション引数に表示条件を入れることで表示を制御することが出来る．

オプション引数の only 2 とは，2 枚目のレイヤーのみで表示される，ということを表している．

```
+multiframe(2){タイトル}<
  +p{1 枚目と 2 枚目で表示される段落．}
  +p?:(only 2){2 枚目のみで表示される処理．}
>
```

挙動を変えるためのコマンド一覧 1

今までに紹介したコマンドのうち，\emph などいくつかのものはオプション引数によってオーバーレイに対応することができる．たとえば \emph では，オプション引数に入れた表示条件を満たした場合のみその文字が強調される．他のコマンドも同様に「表示条件を満たした場合のみ効果が発動する」ようになっている．

- \emph?:(cond){ text }
- \text-color?:(cond)(color){ text }
- \uline?:(cond){ text }
- \strike?:(cond){ text }
- \ctx?:(cond)(cfx-func){ text }
- +ctx?:(cond)(cfx-func){ text }

挙動を変えるためのコマンド一覧 2

特定のスライドで書式を変えるだけでなく，特定のスライドだけ表示したい，非表示にしたいということもよくある．以下のコマンドでそういった表示そのものを制御できる．

コマンド	役割
`+p?:(cond){ text }`	条件を満たすときのみ表示される段落
`+ghost(cond)< block >`	条件を満たすときのみ表示されるブロック
`\ghost(cond){ text }`	条件を満たすときのみ表示されるテキスト
`+phantom(cond)< block >`	条件を満たすときは普通に表示され，満たさないときは同等の空白が表示されるブロック
`\phantom(cond){ text }`	+phantomのインラインテキスト版
`+select(cond)< block >< block >`	条件を満たすときは 2 番目，満たさないときは 3 番目の引数に置き換わるブロック
`\select(cond){ text }{ text }`	+selectのインラインテキスト版

レイヤーの表示条件を定める関数一覧

+p のオプション引数などに与える cond は int -> bool 型の関数であれば何でも良いが，簡単のためエイリアス的な関数をいくつか用意している．「m 枚目から n 枚目まで」のように比較的単純な条件であれば以下の関数を用いると楽．

いずれも SlydifiOverlay モジュール中にあるため，最初に open SlydifiOverlay を行うと良い．

関数	挙動
only N	N 番目のレイヤーでのみ成立
elsewhen N	N 番目のレイヤー以外でのみ成立
till N	N 番目のレイヤーまで成立
after N	N 番目のレイヤー以降のみ成立
during M N	M 番目から N 番目のレイヤーでのみ成立
in-case [N1; N2; ...]	リストに数字が入っているレイヤーでのみ成立

図版の挿入

FigBox モジュール

スライドというコンテンツでは，通常の文書以上に図版が重要な役割を持つ．図版の大きさ，配置，余白の量がスライドの見た目に大きな影響を与え，結果的に通常の文書よりも自由度の高い設定が求められることになる．

SLγDIF$_{\mathrm{I}}$ では，カスタマイズしやすい図版の挿入を実現するために `FigBox` モジュールを設け，独自のインターフェースを実装している．

以下では，プリアンブルにて `FigBox` モジュールを `open` しているものとする．

```
open FigBox
```

図の配置コマンド

FigBox.t 型の値 figbox を指定して，以下のコマンドを用いることで図が挿入される．

コマンド	役割
+fig-center(figbox);	中央揃えで配置
+fig-block?:(align)(figbox);	align で指定した揃え方で表示（左右揃え）
+fig-on-right(figbox)< block >;	本文 (block) の横幅を figbox の横幅の分だけ短くし，本文の右側に図を表示．
+fig-on-left(figbox)< block >;	本文 (block) の横幅を figbox の横幅の分だけ短くし，本文の左側に図を表示．
+fig-abs-pos((x, y))(figbox);	絶対座標で配置
+fig-inline(figbox);	インラインテキスト中に図を挿入

「図」として配置可能なもの

FigBox.t 型のコンテンツは以下のようにして作成する．つまり，以下が「図」として配置可能なものの一覧である．

コンテンツ	関数
JPEG 画像	`include-image width `path/to/file.jpg``
PDF （1 ページ目）	`include-image width `path/to/file.pdf``
PDF （nページ目）	`include-image ?:n-page width `path/to/file.pdf``
JPEG 画像 （縦幅指定）	`include-image-with-height height `path/to/file.jpg``
ダミーのブロック	`dummy-box width height`
インラインテキスト	`textbox ?:(ctx-func) { text }`
インラインテキスト （幅指定）	`textbox-with-width ?:(ctx-func) width { text }`
ブロックテキスト	`textblock ?:(ctx-func) width '< text >`

図の結合

複数の図を並べたい，という需要は少なくない．SLγDIFl では複数の図を簡単に並べることができる．

内容	関数
図を横に並べる	`hconcat ?:(align) [ figbox-1; ...; figbox-n ]`
図を縦に並べる	`vconcat ?:(align) [ figbox-1; ...; figbox-n ]`

- オプション引数 `align` に `float` 型の値を指定することで，揃え方を変更できる．
  - `0.0`: 左揃え / 下揃え
  - `0.5`: 中央揃え
  - `1.0`: 右揃え / 上揃え
- `gap` 関数を途中に挟むと，指定したぶんの余白が図と図の間に挿入される．
  - `hconcat` でも `vconcat` でも使える
  - 例： `hconcat [ figbox-1; gap 10pt; figbox-2; gap 5pt; figbox-3 ]`

図の変換

「縦に並べる」「指定した大きさのマージンを周囲に足す」といった操作を図に加えたいこともある．SLγDIFI では figbox を入力として新たな figbox を返す関数をいくつか用意しており，既存の図から簡単にマージンなどの要素を加えた新たな図を作成できる．

変換内容	関数
左右に余白を追加	`figbox |> hmargin length`
上下に余白を追加	`figbox |> vmargin length`
上下左右に余白を追加	`figbox |> hvmargin length`
上下左右に異なる大きさの余白を追加	`figbox |> margin left right top bottom`
フレーム （外枠） を描画	`figbox |> frame line-width line-color`
図の背景色を設定	`figbox |> bgcolor color`

- 複数の変換を組み合わせることも当然可能．
  - 例： `include-image 100pt` `` `file.pdf` `` `|> hvmargin 10pt |> frame 1pt Color.black`
- `hconcat` や `vconcat` と組み合わせることも当然可能．

図のオーバーレイ表示

図もオーバーレイに対応しており，特定のレイヤーに応じて図を出し分けることが可能.

表示条件を満たすときに図を表示し，それ以外のときは大きさを 0 にしたいときは ghost 関数を用いて figbox |> ghost（cond）とする.

表示条件を満たすときに図を表示し，それ以外のときは図を不可視としたい（すなわち，元の図と同じ大きさの透明の箱を置きたい）ときは phantom 関数を用いて figbox |> phantom（cond）とする.

表示条件に応じて図を出し分けたいときは，select 関数を用いて select（cond）figbox-true figbox-false とする.