卒業論文

hikiutils を用いた
卒業論文作成

関西学院大学　理工学部　情報科学科

1234 西谷滋人

2017 年 3 月

指導教員　　西谷　滋人　教授

目次

1 イントロダクション

卒業論文は，大学で課される普通の授業のレポートに比べて多くの情報内容が要求されます．また，研究という側面から体裁や版権など多くの出版における掟を遵守しながら，高い品質を保つ必要です．そのため，指導教官との編集作業が重要となるます．編集作業の効率を高めるためには，共同作業を促進するプラットフォームが必要です．

体裁などを考えると Latex の使用が標準ですが，執筆段階での煩わしさを低減するため，Mark Up 記法による文章作成が一般的です (「ドキュメント作成システム構築ガイド，GitHub, RedPen, Asciidoctor, CI によるモダンライティング」, 伊藤敬彦，吉村孝広，技術評論社)．西谷研では Mark Up 記法の一種である hiki 記法を用いて執筆します．hiki 記法から html へ変換するソフト hikidoc による容易な変換が可能なため，日記 web appli の hiki diary でも利用されている有名なソフトです．hiki 記法をベースにして，wiki wiki web と同等の機能を提供する hiki システムが github に公開されており，個人の手持ちのパソコンにインストールして wiki を利用することができます．西谷研ではさらに hiki 記法を拡張して，コードのカラー表示，数式の latex 記述が可能となるシステムを利用しています．

本資料では，卒論の編集作業を効率化する図に示すようなシステムの使い方を紹介します．

例えば，卒業生を yamane としましょう．yamane の個人の Mac の自分の directory(hikiutils_yamane) に幾つかのファイルを作成して卒論を書いているとします．これを指導教官 (bob) が編集するとします．この同期には，Github により提供される共同作業環境を使います．これだけでは編集中の文書の体裁がわかりにくいでしょう．そこで，hiki システムにより容易に web ブラウザ上に完成形を表示しつつ執筆することが求められます．このような操作環境を提供するのが，hikiutils -i です．

2 hiki との同期

2.1 hikiutils の install

hikiuitls を gem から install しておく必要がある．コマンドは以下の通り．

```
gem install hikiutils
```

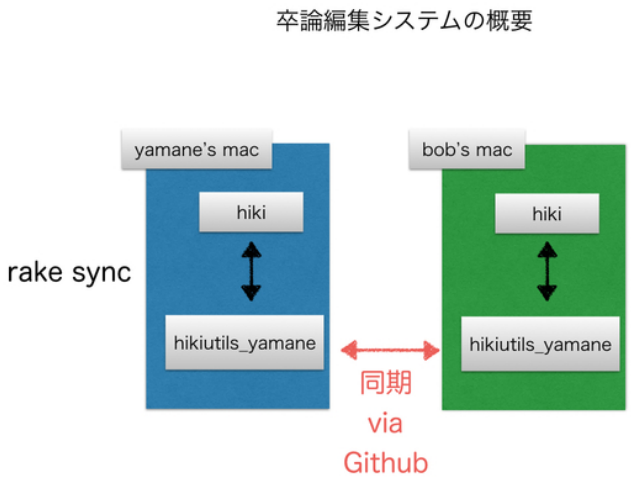

図 1　卒論編集システムの概要.

さらに

```
hiki -v
```

で 0.2.3.2 以上であることを確認.

2.2　個別ディレクトリーの構成

図に hikiutils_bob のディレクトリー構成を示す.
コマンド

```
hiki -i
```

によって以下のようなファイルが作成される.

```
1 bob% ls -lat
2 total 1072
3 -rw-r--r--   1 bob  501     99  1 20 12:44 .hikirc
4 drwxr-xr-x   5 bob  501    170  1 20 12:44 figs/
5 drwxr-xr-x  12 bob  501    408  1 20 12:34 ./
6 -rw-r--r--   1 bob  501      8  1 20 11:01 .gitignore
```

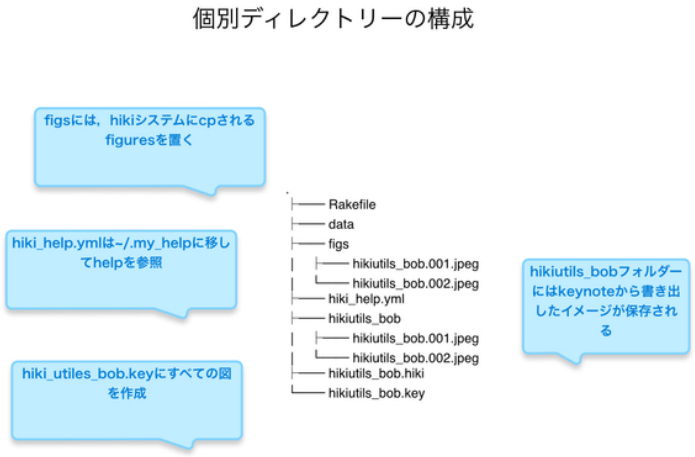

図 2　hikiutils_bob のディレクトリー構成.

```
−rw−r−−r−−   1 bob  501    3507  1 20 11:01 Rakefile
drwxr−xr−x    2 bob  501      68  1 20 11:01 data/
−rw−r−−r−−   1 bob  501    2595  1 20 11:01 hiki_help.yml
drwxr−xr−x   26 bob  501     884  1 20 11:00 ../
```

この.hikirc にデータが設定データが自動的に入る．さらに hikiutils_bob.hiki および hikiutils_bob.key を作成する．これで執筆ファイルの基本構成が出来上がる．

keynote で図を作成して，hikiutils_bob.hiki に文章を記述していく．

2.3　一般的な執筆手順

1. 書類の作成
   (a) open -a mi hikiutils_bob.hiki
2. keynote を開ける
   (a) open hikiutils_bob.key
3. keynote のイメージを figs に
   (a) keynote でイメージへ書き出し (hikiutils_bob を仮定)
   (b) rake convert 80 hikiutils_bob
4. hiki システムとの同期

(a) rake sync

5. hiki システムで表示
   (a) hiki -u hikiutils_bob

2.4 rake が用意しているタスク

rake の用意しているコマンドは次のとおり.

```
1 rake check_previous  # check previou and current sync
2 rake chenv           # For hiki Errno::ENOENT, Errno::
      EACCES
3 rake convert         # convert fig size SCALE TARGET_DIR
4 rake force_sync      # force_sync hikis and figs to hiki
      directory
5 rake increment       # increment fig NUBERS in FILE
6 rake number          # numbering figs from the NUBER in
      FILE
7 rake self_copy       # self copy to hikiutils template
      directory
8 rake sync            # normal sync hikis and figs to hiki
       directory
9 rake sync0           # sync0 hikis and figs to hiki
      directory
```

2.4.1 rake sync

hikiutils_bob にある必要な書類を hiki システムにコピーする．その際，名前の書き換えを行う.

表 1

hikiutils_bob での名前	hiki システムでの名前
hikiutils_bob.hiki	hikiutils_bob
introduction.hiki	hikiutils_bob_introduction

figs ディレクトリー内のファイルは hiki/cache/attache/hikiutils_bob に cp される．従って，hiki 文書中で参照するには，`{{attach_view(hogehoge.png)}}` という記述が

必要となる．

2.4.2 rake convert

keynote が吐き出したイメージを変換するためのコマンド．ImageMagick がインストールされている必要がある．ない場合は，自分で brew から install するか，`https://www.imagemagick.org/script/binary-releases.php` からダウンロード．うまくいかない場合は donkey に聞いてみてください．

```
rake convert 80 hikiutils_bob
```

によって，hikiutils_bob に keynote から吐いた png ファイルを 80

2.4.3 rake force_sync

hiki システム側で直接変更を加えると，hikiutils が sync した時と差ができる．これを検知して，ユーザに注意を喚起する仕組みがある（rake check_previous)．これは sync した時に自動的に呼び出される．違いの出た files を修正した後に強制的に同期をとるためのコマンドとして，force_sync が用意されている．

2.4.4 rake chenv

hiki システム上で error が出た場合に試してほしい．error の状況は個人の設定によってちがうため，対処法の実装は網羅されていない．うまくいかない場合は西谷に Issues として投げるように．

2.4.5 rake increment

keynote でページを追加すると hiki での参照 (attach_view) にずれが生じる．いまのところこれを解消する方法はなく手で修正を加える必要がある．ずれが単純な場合には，

```
1 cp hikiutils_bob.hiki tmp.hiki
2 rake increment 2 tmp.hiki > tmp2.hiki
```

として attach_view のページ番号を単純に増加させることができる．

2.4.6 rake number

前節同じく figs 内の通し番号が変わった時に attach_view の通し番号を調整するコマンド．

```
rake number 3 hikiutils_bob.hiki > tmp.hiki
cp tmp.hiki hikiutils_bob.hiki
```

とすると

```
8c8
{{attach_view(hikiutils_bob.002.jpeg)}}
---
{{attach_view(hikiutils_bob.003.jpeg)}}
21c21
{{attach_view(hikiutils_bob.003.jpeg)}}
---
{{attach_view(hikiutils_bob.004.jpeg)}}
```

などと番号を 3 から順に振り替えてくれる.

2.5 github による同期

1. git init が済んでいると仮定
2. upstream, origin の確認
   (a) bob
3. git push 作業
   (a) git add -A
   (b) git commit -m 'first commit'
   (c) git push origin master
4. github で bob へ pull request をかける
5. 編集後
6. git pull 作業
   (a) git pull upstream origin

これがうまくいかん時は聞いてください.

hikiutil 関連のヘルプ

- hikiutil 関連のヘルプ

2.6 hiki_help の出力

2.6.1 hiki で卒論を書くときの初期化と掟

- 開発メモ:figs,data も作成
- 目的：西谷が後で迷わないように決まったファイル構造を堅持すべし
- 文書：hiki で書く. のちには，latex に変換するプログラムを提供します
- 図表：すべての図表を keynote にまとめる，タイトルを分かりやすく書く
- データ：data ディレクトリにまとめる．ファイル名を keynote の対応する図表中に記す
- hiki –initialize で初期ファイル (Rakefile, ./.hikirc, hiki_help.yml) が copy される
- hiki_help.yml を適宜 /.my_help に copy して hiki_help として利用，(my_help 参照)
- Errno::EACCES や permission error がでたときは rake chenv を試してみる（報告して）
- rake sync によって hiki ディレクトリーと同期が取られる
- hiki -u TARGET によってブラウザー表示される
- テキストの拡張子は'.hiki'
- hiki での url はテキスト前とディレクトリーから自動生成される
- 例えば，hiki2latex_saki/introduciton.hiki とすると hiki2latex_saki_introduciton と変換される

2.6.2 error 対応

- Permission denied - ./data/text/boundary_narita (Errno::EACCES)-¿テキストに hiki が書き込めない，chmod a+w FILE

2.6.3 図表：すべての図表を keynote にまとめる，タイトルを分かりやすく書く

- keynote に書いたスライドはイメージに書き出して，rake convert 80 TARGET_DIR で figs に変換
- rake sync で figs にある files は hiki/target_dir に cp される
- `convert #{source} -resize 20% #{target}` **によって,** `target=figs/TAERGET.png` **に** 20% **に縮小して保存される**

- convert -density 300 view.svg view.png で 300dpi で変換
- attach_anchor では

'`{{attach_anchor(test.png, hiki2latex_saki)}}`' と，directory 指定しなければならない.

- keynote であとで図を挿入して番号が変わった時の原稿の一括変換
- rake increment 2 boundary_bob.hiki boundary_bob ¿ tmp.hiki
- rake convert 60 boundary_bob
- rake sync
- hiki -u boundary_bob_tmp

3　実際の最終形態 (卒論=pdf) への変換

hiki+keynote で卒論の内容ができたら，それを卒論の最終形態つまり pdf へ変換する必要が出てきます．

ここで問題が発生します．hiki よりも latex の方が機能が豊富なこと．つまり，簡易に書くには hiki などの mark down で書いていくのがいいんですが，文書として完成度を高めるには，latex で細かい設定を調整する必要がどうしても出てきます．

具体的には，

1. 図の配置を調整する wrap の数値調整
2. 参照文献の記述
3. リスト，引用の体裁
4. 章立て階層構造

などが問題になるところです．

これは hiki ではどうしようもありません．一つの手はもう少し高機能な mark up 言語例えば asciidoc などに変更することですが，これはどこまでいっても終わりのない方向のようで，結局は latex で書いているのと同じになる可能性があります．

我々は違う戦略をとります．それは，

`latex` をベースにして，`hiki` を生成する

と言う手です．

DRY(Don't Repeat Yourself) 原則さえ守れば，文書管理はいいのですから，ある段階まですすめば hiki ではなく latex を原本にするのです．そのための変換器 latex2hiki とその派生 rake 環境が用意されています．そちらはこちらで卒業後に変換します．参照文献は latex に移してから修正してください．

ここでは，hiki でできることをとことん突き詰めておきます．

3.1　install

```
hiki -i
```

で install されています．新たに使うコマンド群は次の通りです．

```
1 rake change_wrap      # change latex figures to wrap
```

```
    format
2 rake latex            # latex conversion FILE1
3 rake latex_all        # latex conversion whole hiki files
    in the current dir
4 rake latex_base       # latex conversion FILE1(hiki) to
    FILE2(latex)
5 rake latex_wrap       # latex conversion FILE1 with wrap
    format
```

3.2 rake latex 手順（個別ファイルの変換）

```
rake latex sync.hiki
```

とすると，

```
latex_dir/sync.tex
```

に latex 変換後のファイルが生成します．これを，TeXShop で command_t 変換します．

- {{attach_anchor(sync.pdf,hikiutils_bob)}}

うまくいかないときは，terminal で

```
platex sync.tex
dvipdfmx sync.dvi
```

を試してみてください．

3.2.1 注意

hiki の初めの部分は，

```
!title:hikiutils -i による卒論作成システム
!author:Shigeto R. Nishitani
!date: Kwansei Gakuen Univ., 2017/1
```

とすると，

```
1 \begin{document}
2 \title{hikiutils −による卒論作成システムi}
```

```
3 \author{Shigeto R. Nishitani}
4 \date{ Kwansei Gakuen Univ., 2017/1}
5 \maketitle
```

と変換してくれます.

必要な図は，figs から自動で取るようになっていますが，サイズや解像度が問題のときは手動で調整してください.

3.3 rake latex_all 手順（ディレクトリー内の一括変換）

pwd の directory と同名の basename に.hiki の拡張子がついたファイルが用意されていて,

```
rake latex_all
```

をおこなうと自動で全部を一体にまとめた文書への latex 変換が出来上がります．たとえば，hikiutils_bob.hiki の記述を

```
bob% cat hikiutils_bob.hiki
!hikiutils を用いた卒業論文作成


! [[hikiutils_bob_sync]]
! [[hikiutils_bob_latex_all]]
```

とすると，sync.tex, latex_all.tex が latex_dir 内に変換されます．さらに，開いている hikiutils_bob.tex を TeXShop で変換してみてください.

うまくできないときは**ここ** (http://qiita.com/hideaki_polisci/items/3afd204449c6cdd995c9
を参照して自力で入れてみてください．だめなら donkey.

- {{attach_anchor(hikiutils_bob.pdf,hikiutils_bob)}}

というようになります.

3.3.1 下準備

latex_dir 内に幾つかの tex 雛形を入れておく必要があります.

```
hiki -i
```

と hikiutils 環境を再度初期化すると，自動でインストールする設定です．なかったら手動で作ってください．

■head.tex

```
bob% cat head.tex
\title卒業論文{\\
\vspace{4cm} を用いたhikiutils卒業論文作成\\}
\author{ 関西学院大学　理工学部　情報科学科\\\\1234 西谷滋人}
\date{\vspace{3cm} 年2017 月 3\\
\vspace{3cm} 指導教員　　西谷　滋人　教授}
\maketitle
```

■pre.tex

```
bob% cat pre.tex
\documentclass[10pt,a4j]{article}

\def\Vec#1{\mbox{\boldmath $#1$}}
\usepackage[dvipdfmx]{graphicx}

\setlength{\textheight}{275mm}
\headheight 5mm
\topmargin -20mm
\textwidth 160mm
\textheight 250mm
\oddsidemargin -0mm
\evensidemargin -5mm

\pagestyle{empty}
\makeatletter
  \def\@maketitle{%
  \newpage\null
  \vskip 2em%
  \begin{center}%
  \let\footnote\thanks
    {\large\bf \@title \par}%
    \vskip 1.5em%
    {\large\bf \@author \par}%
```

```
25      \vskip 1.5em%
26      {\small \@date}%
27    \end{center}%
28 }
29 \makeatother
30
31 %\documentclass[10pt, a4j]{article}
32 %%\usepackage{citesort}
33 \usepackage{amssymb}
34 \usepackage[dvipdfmx]{graphicx}% 図を入れるときに使用
35 \usepackage{wrapfig}% 図の周りに本文を流し込みたいときに使用
36 \usepackage{subfigure}
37 \usepackage{here}
```