

卒業論文

ユーザメモと wiki を連携するシステムの開発

関西学院大学 理工学部 情報科学科

3550 江本 沙紀

2017 年 3 月

指導教員 西谷 滋人 教授

目次

| | | |
|-----|---------------------------------|----|
| 1 | 開発の背景 | 3 |
| 2 | 関連するソフトの振る舞い | 5 |
| 2.1 | my_help | 5 |
| 2.2 | hiki | 6 |
| 3 | 方法 | 8 |
| 3.1 | 使用法 , コマンド | 8 |
| 3.2 | TARGET -push のコードの詳述 | 9 |
| 3.3 | my_help -hiki のコードの詳述 | 11 |
| 4 | FrontPage の設計 | 13 |
| 5 | 結果 | 16 |
| 6 | 今後の課題 | 19 |

1 開発の背景

近年、ナレッジマネジメントが企業経営の重要な要素と言われ、導入する企業が増えている。ナレッジマネジメントとは、個人の持つ知識や情報を組織全体で共有し、有効に活用することで業績を上げようという経営手法である。日本語では、「知識管理」などと訳され、「KM」と略されることもある [1]。

ナレッジマネジメントでは、グループ開発において共有する知識は暗黙知と形式知に分けられる [2]。暗黙知は主に口伝によって一対一でつたえられたり、あるいは体で覚えるというのが一般的である。しかし、定着するまでの間は一般的にメモという形で個人的な知識として扱われるのが普通である。一方で図書館や web などでは文書や hyper text として誰もが読める形で保管、提供される知識は形式知と呼ばれる。

形式知と暗黙知

| 形式知 | 暗黙知 |
|----------------------------|-------------------------------|
| 言葉や文字で表現できる 記録として残されている | 言葉や文字で表現できない 無意識や筋肉記憶の中に存在 |
| 多くの人に伝えるときに効果的 | 人と人のコミュニケーションに よってのみ伝達可能 |
| (例)料理本, 教科書 | (例)包丁の使い方, 火加減 |

図 1 暗黙知と形式知

暗黙知の形式知化はいくつも行われており、google 検索したときによく見る Qiita.com などそれらをまとめるサイトを提供している。西谷研究室では、各所属学生の暗黙知を形式化するために、my_help という gem を開発し、自分のためのメモを残して活用している。本研究では、西谷研究室内でのナレッジマネジメントを推進するため、メモソフト my_help から wiki clone の hiki へ自動変換するシステムの開発と、my_help をよりよい

ソフトにするために FrontPage の設計をする .

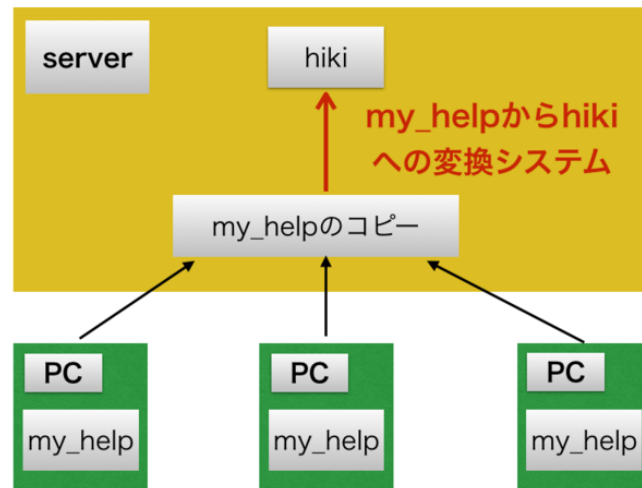


図 2

本論文の構成は次の通りである．対象とするソフトはあまり一般に普及しているものではないため，最初にソフトの特徴と簡単な振る舞いを紹介する．次に 3 章では作成したソフトの振る舞いとコードの中身を詳述する．4 章では今後これらの変換ソフトを使って web を作成するためにプロトタイプを作成し，そこで必要とされる機能について検討を行っている．5 章では結論をまとめ，最終章で今後の課題を記した．

2 関連するソフトの振る舞い

本章では、対象とするソフトはあまり一般に普及しているものではないため、最初にソフト (`my_help` および `hiki`) の特徴と簡単な振る舞いを紹介する。

2.1 `my_help`

`my_help` は西谷研究室で使用しているユーザ独自のメモを作成する `gem` である。

`gem` は正式名称を `RubyGems` といい、Ruby 用のライブラリを使う時に必要となるソフトウェアのことである [3]。パッケージ管理ツール `gem` があることで、Ruby 用ライブラリのインストール、アンインストール、バージョン管理などを簡単に行うことができる。プログラミング言語 Ruby のファイルに付属されていて、無料で利用することができる。`gem` の利点は次の通りである。

- 標準化された構造があるので、初めてみた人でも分かるようになっている。
- `gem` があることで、簡単に Ruby 用ライブラリをインストールでき、初心者でもアプリ機能を装備できる。
- 誰でも作成、配布が可能である。

`my_help` が提供するコマンドとその振る舞いを `help` 表示から示す。

```
Usage: my_help [options]
  -v, --version                show program Version.
  -l, --list                   list specific helps
  -e, --edit NAME              edit NAME help(eg test_help)
  -i, --init NAME              initialize NAME help(eg test_help).
  -m, --make                   make executables for all helps.
  -c, --clean                  clean up exe dir.
  --install_local              install local after edit helps
  --delete NAME                delete NAME help
  --hiki                       my_help2hiki
```

`my_help` では `help` の要素をこの `help` 表示と同じ構造、項目と対応する記述で表示される。それぞれの項目はまとまりをつくり、それを `-l` でリスト表示される。`-version` お

よび`-list` は `gem` 標準に準拠するために用意されている．上記コマンドの `help` 表示の `NAME` には作成した `help` の名前が入る．`-edit NAME` により，`NAME` の内容を記述し更新する，`-init NAME` により，`NAME` を初期化する．`exe` のディレクトリを空にするには`-clean` のコマンドで実行ができる．`help` を書いた後に `my_help` によって記述や閲覧ができるようにするために，`-install_local` が用意されている．`NAME` の `help` を消したいときには，`-delete NAME` によって消去ができる．`-hiki` コマンドにより，`my_help` から `hiki` への自動変換を行い，`wiki` でブラウザ表示を行う．`my_help` を研究室内で利用する利点は次の通りである．

- 研究室でのメモの書き方が統一できる．
- どこにメモをしたか忘れることがない．
- 普段研究の為に使うターミナルから離れることなくメモを残すことができるので，書きたいときにすぐ書くことができる．

2.2 hiki

`hiki` とは，`Ruby` で書かれた高機能・高速 Wiki クローンである [4]．`CGI`(`Commond Gateway Interface`) を利用して，`Web` サーバと連動して動く [5]．西谷研究室では，`hiki` の形式を利用してサイトを作り，研究室での情報共有や `gem` の使い方などを掲載して閲覧できるようにしている．また，卒業論文の作成も `hiki` の形式で作成している．`hiki` の特徴として次のことが挙げられる．

- オリジナル Wiki に似たシンプルな書式を持つ．
- プラグインによる機能拡張ができる [6]．
- 携帯からアクセスが可能．
- アクセス制限ができる．
- 柔軟性が高く，手軽に始められて操作が簡単 [7]．
- 閲覧者でも修正，ページの追加などの編集が行える．
- 作成したページを自動で整理する [8]．

これらの特徴から，複数の人が同じ情報をリアルタイムで共有し，編集しやすいという利点を持つ．まとめサイトを作るのに適しているとして，多くのサイトに使われている．代表的なものに百科事典である `Wikipedia` がある．本研究で作成を行う西谷研究室の内部サイトも，各研究室メンバーのメモ（暗黙知を形式知化したもの）のまとめサイトであ

り，hiki を本研究に利用することは最適であるといえる．

3 方法

本研究で開発した my_help から hiki への自動変換ソフト my_help2hiki は my_help の gem に組み込んでいる．本章では，my_help2hiki のコマンドとコマンドによる振る舞いを記述する．

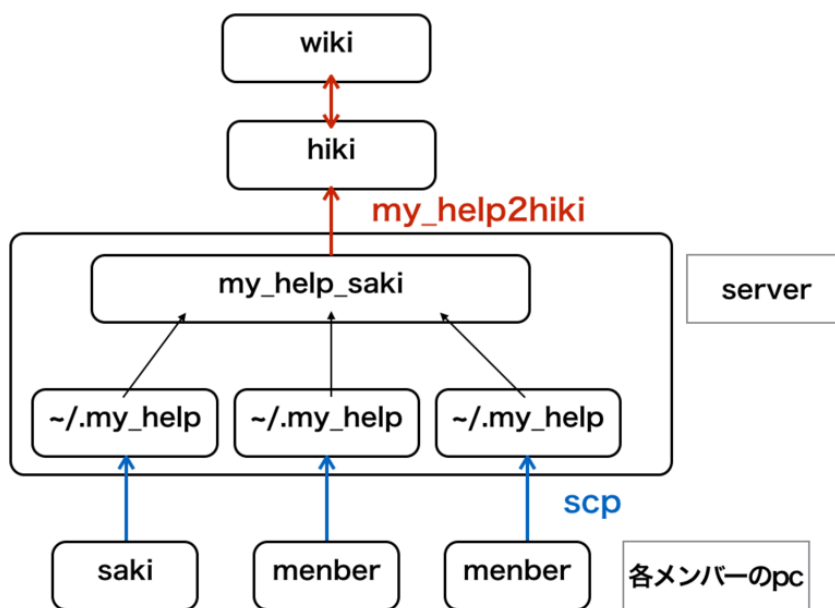


図 3 my_help2hiki

my_help2hiki は図のようになっている，各学生が my_help で作ったメモは my_help のディレクトリに保存される．これを scp コマンドを利用して server の my_help_saki のディレクトリにコピーする．コピーした my_help の内容を my_help2hiki を利用して hiki に変換し，wiki で表示する．

3.1 使用法，コマンド

my_help から hiki に変換するときのコマンドと使用法は以下の通りである．

- TARGET -push : 作成したメモ (TARGET) をサーバに送る
- my_help -hiki : 作成したメモを hiki 形式に変換し，wiki で表示できるようにする．

3.2 TARGET -push のコードの詳述

TARGET -push のコードについて詳述する .

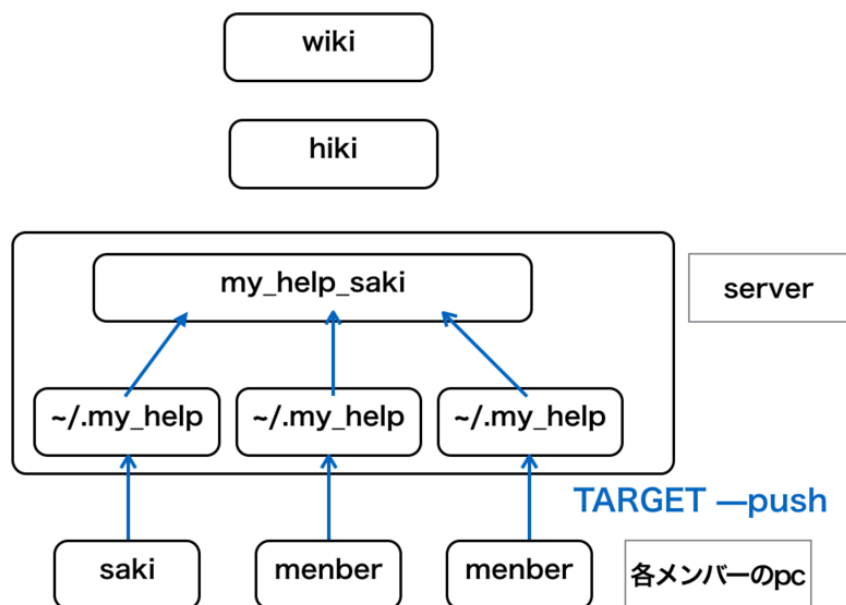


図 4 TARGET -push

コードの中身は以下の通りである .

```
def push
  p "push my_todo"
  data_dir = File.join(ENV['HOME'], '.my_help')
  FileUtils.cd(data_dir)
  system "pwd"
  system "rm -rf ~/.my_help/*.yaml~"
  system "scp -r ~/.my_help saki@nishitani0:~"
  system "ssh saki@nishitani0 ls ~/.my_help"
end
```

コードの詳細を記述する．

- 3,4 行目

`my_help` では，作成したメモが `my_help` のディレクトリに自動的に追加されるので，ディレクトリを `my_help` に移動する．

- 6 行目

`my_help` にメモが追加されるとき，`yml` 形式のファイルで保存される．メモを更新すると，一つ前に保存したファイルは `*.yml` というファイル名でバックアップとして残される．`rm -rf` で unnecessary ファイルは削除し，サーバにコピーするときのデータ量を減らしている．

- 7 行目

`scp -r /[directory 名] [server 名]` のコマンドによって `server` に `ssh` 接続を行い，`directory` を `server` にコピーする．`-r` はディレクトリ全体をコピーすることを示している．西谷研究室で利用している `nishitani0` というサーバにコピーしている．

- 8 行目

`nishitani0` に `ssh` 接続し `my_help` の中身を書き出して，コピーができているかコマンドを実行した時に確認が行えるようにしている．

3.3 my_help -hiki のコードの詳述

my_help -hik のコードについて詳述する .

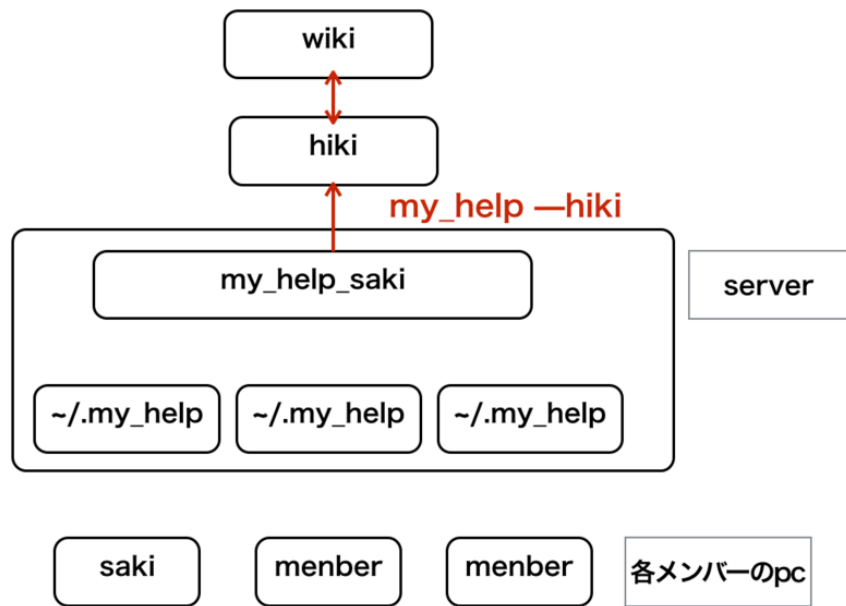


図 5 my_help2hiki

コードの中身は以下の通りである .

```
def hiki
  p 'my_help2hiki'
  system "emacs_help --to_hiki > ~/Sites/hiki-1.0/data/text/emacs_help_saki"
  system "my_todo --to_hiki > ~/Sites/hiki-1.0/data/text/my_todo_saki"
  system "ssh_help --to_hiki > ~/Sites/hiki-1.0/data/text/ssh_help_saki"
  system "open -a safari 'http://localhost/~saki/hiki-1.0/?FrontPage'"
end
```

コードの詳細について記述する．

- 2-4 行目

my_help には，**TARGET -to_hiki** というコマンドがあり，これによって yaml 形式で保存されているメモを hiki 形式で書き出すことができる．この **-to_hiki** のコマンドを使って hiki 形式にしたものを，wiki で表示することのできるフォルダである **/Sites/hiki-1.0/data/text/** に入れることで，wiki での表示を可能にしている．emacs_help，my_todo，ssh_help は全て私の my_help に入っているメモ．

- 5 行目

wiki のページである，図 6 に示した FrontPage を表示するコマンド．これによりメモが更新されているのをすぐに確認することができる．FrontPage は以下のようになっている．

```
!saki's help
*[[ssh_help_saki]]
*[[my_todo_saki]]
*[[emacs_help_saki]]
```

先頭に!をつけることで 1 行目の saki's help を見出しにし，2 4 行目は*によって箇条書き，角括弧でリンクになっている．



図 6 コマンドを実行したときに開く FrontPage

4 FrontPage の設計

my_help をよりよくするための設計を示す．各学生の FrontPage , 各 help のページに分けて，実装すると便利になる理由と共に記述している．

my_help2hiki_saki_prototype

saki's help

- [ssh_help_saki](#)
- [my_todo_saki](#)
- [emacs_help_saki](#)

①メモの内容
による分類

nishitani group helps

- [yamane_help?](#)
- [nasu_help?](#)
- [nari_help?](#)

②研究室内
メンバーの
helpへのリンク

top ranked helps

| rank | dl | link | description |
|------|----|-------------------------------------|-------------|
| 1 | 12 | ssh_help_yamane? | sshの使い方 |
| 2 | 4 | github_help_yamane? | githubの基本 |

③閲覧回数が
上位のhelp
へのリンク

new helps

| date | link | description |
|------|-----------------------------------|-------------|
| 2/6 | gnuplot_help_bob? | gnuplotの使い方 |

④更新の
新しいhelp
へのリンク

図 7 FrontPage

● 1. メモの内容による分類

研究室内の所属学生の利用するハードウェア，ソフトウェアは同じものが多い．例えば，西谷研究室ではハードウェアは全員が mac を使い，プログラム作成はターミナル，hiki 文書の作成には mi というソフトを使用している．それぞれのハードウェア，ソフトウェアに関する help を分類分けしておくことで，調べたいことに関する help を探しやすくすることができる．

- 2. 研究室内のメンバーの help へのリンク

同研究室の他学生の FrontPage へのリンクを作る．同回のメンバーが書いたメモを見たい，先輩のメモを見たい，他メンバーの研究を知りたい．そのときの目的に応じて閲覧する help を選ぶことができる．

- 3. 閲覧回数が上位の help へのリンク

閲覧回数の多い，研究室内の学生が見た回数の多いものへのリンクを作る．研究室に入っただけで分からないことが多いときにこの help を見れば，研究室のことが理解できる．知らなくても支障はないが，知っておくと便利な豆知識を得ることが期待される．

- 4. 更新の新しい help へのリンク

更新が新しいものを表示しておくことで，メンバーが得た最新の知識を得やすくなる．また，他メンバーがどのような研究を進めているか，どのようなことを調べてたのかを知ることができ，自分の研究の進め方の参考にすることができる．

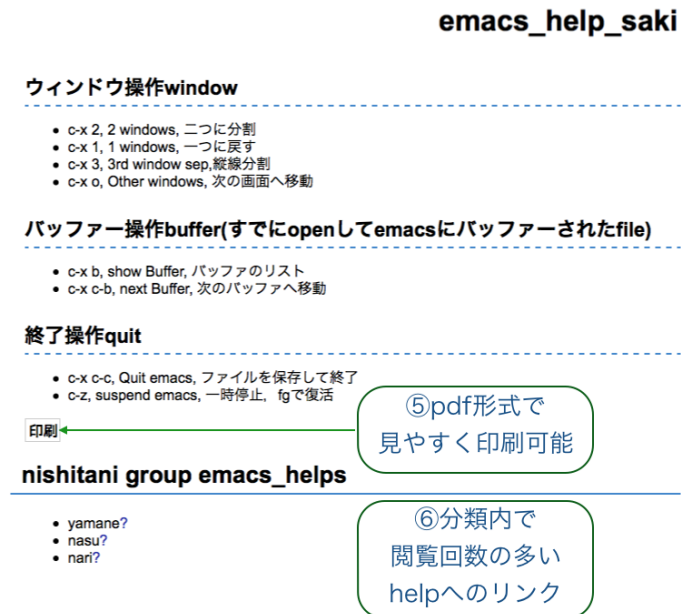


図 8 例 emacs.help から一部抜粋

● 5. pdf 形式で見やすく印刷可能

紙媒体で持ち歩くことを可能にするために、pdf 形式に変換することで見やすくした help を印刷することができる。emacs.help を例にして考える。この help はターミナルで emacs を使うときのキーバインドを表している。ターミナルを利用していると、この help を開きながら操作することは手間がかかる。そこで見やすくした help を紙に印刷することでこの手間を省くことができる。西谷研究室では hiki から latex(pdf 形式への変換を可能にするソフト) への変換システム、hiki2latex があるので、作成可能だと考えられる。

● 6. 分類内で閲覧回数の多い help へのリンク

分類内で閲覧回数の多い help は他メンバーの多くが得た知識なので、知っておくべき知識であるといえる。自分がその分類内で分からないことを解決する手がかりになることが期待される

5 結果

```
/Users/saki% my_help --list
Specific help file:
  emacs_help :emacs のキーバインド
  memo_help  :ヘルプのサンプル雛形
  my_todo    :my_todo
  ssh_help   :ssh の help
```

これは私の my_help の中身を書き出している．下が emacs_help の中身である．

```
/Users/saki% emacs_help --all
emacs のキーバインド
```

特殊キー操作

```
  c-f, control キーを押しながら      'f'
  M-f, esc キーを押した後一度離して'f'
      操作の中断 c-g, 操作の取り消し (Undo) c-x u
      cc by Shigeto R. Nishitani, 2016
+emacs のキーバインド:
+
特殊キー操作
+  c-f, control キーを押しながら      'f'
+  M-f, esc キーを押した後一度離して'f'
+      操作の中断 c-g, 操作の取り消し (Undo) c-x u
+      cc by Shigeto R. Nishitani, 2016:
---
```

-カーソル移動 cursor:

```
+c-f, move Forward,      前 or 右へ
+c-b, move Backward,     後 or 左へ
+c-a, go Ahead of line,  行頭へ
+c-e, go End of line,    行末へ
+c-n, move Next line,    次行へ
```



```

+c-p, move Previous line, 前行へ
---
---
-ページ移動 page:
+c-v, move Vertical,          次のページへ
+M-v, move reversive Vertical, 前のページへ
+c-l, centerise Line,        現在行を中心に
+M-<, move Top of file,      ファイルの先頭へ
+M->, move Bottom of file,   ファイルの最後尾へ
---
---
-ファイル操作 file:
+c-x c-f, Find file, ファイルを開く
+c-x c-s, Save file, ファイルを保存
+c-x c-w, Write file NAME, ファイルを別名で書き込む
---
---
-編集操作 edit:
+c-d, Delete char, 一字削除
+c-k, Kill line,   一行抹消, カット
+c-y, Yank,        ペースト
+c-w, Kill region, 領域抹消, カット
+ 領域選択は, 先頭 or 最後尾で c-space した後, 最後尾 or 先頭へカーソル移動
+c-s, forward incremental Search WORD, 前へ WORD を検索
+c-r, Reverse incremental search WORD, 後へ WORD を検索
+M-x query-replace WORD1 <ret> WORD2: 対話的置換 (y or n で可否選択)
---
---
-ウィンドウ操作 window:
+c-x 2, 2 windows, 二つに分割
+c-x 1, 1 windows, 一つに戻す
+c-x 3, 3rd window sep, 縦線分割
+c-x o, Other windows, 次の画面へ移動

```

```
---  
---  
-バッファ操作 buffer(すでに open して emacs にバッファされた file):  
+c-x b, show Buffer, バッファのリスト  
+c-x c-b, next Buffer, 次のバッファへ移動  
---  
---  
-終了操作 quit:  
+c-x c-c, Quit emacs, ファイルを保存して終了  
+c-z, suspend emacs, 一時停止, fg で復活
```

このように各学生のメモを何種類も作成できるが、自分のパソコンでしか見ることができなかった。本研究による my_help2hiki を使うことで、図 6 のように my_help を wiki で表示可能になり、各学生のメモを研究室内で共有することができるようになる。さらに 4 章で述べたような設計ができれば、研究室内のナレッジマネジメントは今より推進させると考えている。

6 今後の課題

本研究に関する今後の課題として次のことが挙げられる．西谷研究室には内部サイトがあり，研究室内で使うシステムのマニュアルなどが公開されている．hiki 形式への変換ができれば wiki で表示することはできるようになるが，my_help2hiki のコマンドで wiki のページを作成しても，現段階では研究室内全員が見ることはできない．内部サイトへの自動表示を可能にするようなコマンドを作る必要があると考えられる．今後，作成した FrontPage の設計に基づき実装を進めてほしい．

参考文献

- [1] 「e-Words ナレッジマネジメント」, <http://e-words.jp/w/ナレッジマネジメント.html>, 2017/1/27 アクセス.
- [2] ニック・ミルトン, 「プロジェクト・ナレッジ・マネジメント」(生産性出版, 東京都渋谷区渋谷, 2009 年発行), pp.4-5.
- [3] 「Ruby on Rails 初心者必見! パッケージ管理ツール『gem』を徹底解説」, <https://blog.codecamp.jp/rails-gem>, 2017/1/27 アクセス.
- [4] 「Hiki -Front Page-」, <http://hikiwiki.org/ja/>, 2017/1/27 アクセス.
- [5] 「Hiki -Wikipedia」, <https://ja.wikipedia.org/wiki/Hiki>, 2017/1/27 アクセス.
- [6] 「Wiki/Hiki - 180style wiki」, <http://180xz.com/wiki/index.php?Wiki/Hiki>, 2017/2/7 アクセス.
- [7] 「e-Words Wiki」, <http://e-words.jp/w/Wiki.html>, 2017/2/7 アクセス.
- [8] 「Wiki の利点/欠点について検証」, <http://www.sd-dream.com/toolinside/vol078.html>, 2017/2/7 アクセス.