hiki 編集作業を容易にするツールの開発

情報科学科 西谷研究室 3554 山根 亮太

1 目的

hiki は wiki clone として便利.

hiki は，通常 web 上で編集しているが，なにが不便?

GUI と CUI が混在してしまい，操作が不便である．そこで，編集操作が CUI で完結するために，hikiutils が開発された．しかしそのユーザインターフェースに問題がある．現在 hiki でコマンドを入力する際，ユーザーは瞬間的にそのコマンドの機能や意味を判断することが難しく迷いが生じてしまうことがある．

具体的な作業は，hiki のコマンド名を shell 風にすることで瞬時にコマンドの機能を判断し使用することができるようにすることである．

しかし，この開発をすることでユーザーはコマンド入力の際にそのコマンドの機能が分からなくなって入力に迷ったり，コマンド自体を忘れたりすることがなくなりより簡単に入力ができ，時間短縮にも繋がる．そのため，hiki のコマンド名を shell 風に書き換えることでこの問題を解決したい.

2 システムについて

2.1 hiki について

hiki とは wiki エンジンの１つとされており，プログラミング言語 Ruby を用いられることで作られた wiki クローンである．hiki の主な特徴として

- オリジナル wiki に似たシンプルな書式
- プラグインによる機能拡張
- 出力する HTML を柔軟に変更可能
- ページにカテゴリ付けできる
- CSS を使ったテーマ機能
- 携帯端末可能
- InterWiki のサポート
- HikiFarm に対応
- ページの追加，編集がしやすい

等がある [?].

2.2 hikiutils について

hikiutils は，hiki の編集作業を容易に行うことができるようにするツール群である．よって，今回の研究は hikiutils のプログラムを編集することで瞬時にコマンドの機能や意味を理解できるコマンド名を考え編集を行う．hikiutils はプログラミング言語 Ruby のライブラリである gem によって提供されている [?]．hikiutils は CLI(Command Line Interface) で提供されているため，プログラミング言語 Ruby 版のコマンドラインのオプション解析を行いコマンドを標準化にする optparse が利用されている [?]．そのため，CLI の optparse を簡単に扱うために Ruby にはいくつかの library が用意されている．

3 手法

このような CLI の階層的なコマンド実装を示した code として Todo という gem アプリケーションがある [?]．このプログラムでは lib/todo/command/options.rb にサブコマンド用の定義があり，入力されたコマンドは ilb/todo/command.rb で判断されることにより lib/todo/command/options.rb 内で実行される．hikiutils のプログラムの構造は Todo アプリケーションのプログラムの構造と似たものであり hiki の編集を容易にするためのツール群のため，この部分の編集を行うことでコマンド名を書き換える．

4 考察

他の視点から言えない? が見れない?

todo のコードを追いかけて.

直感的なコマンド name？ shell 風，git 風， help?

似たようなソフトに何がある?

似たようなアプリに何がある?

```
hikiutils: provide utilities for helping hiki editing.
"open -a mi"
Usage: hiki [options]
    -v, --version        show program Version.
    -s, --show           show sources
    -a, --add            add sources info
    -t, --target VAL     set target id
    -e, --edit FILE      open file
    -l, --list [FILE]    list files
    -u, --update FILE    update file
    -r, --rsync          rsync files
        --database FILE  read database file
        --display FILE   display converted hikifile
    -c, --checkdb        check database file
        --remove FILE    remove file
        --move FILES     move file1,file2
        --euc FILE       translate file to euc
```

現在 Todo アプリケーションのコードを追いかけて，プログラムの構造を理解している．しかし，hiki のコマンド名の変更や直感的なコマンド名を考えることは現在できていない．今後の課題としては，まず初めに Todo アプリケーションの構造を理解することで hikiutils の構造を理解する．また，誰が見ても機能が判断できるコマンド名を考え，実際に hikiutils 内でそのコマンドの編集を行って実装し，そのコマンドが分かりやすいものであるかどうかを試してもらう必要がある．

参考文献

[1]「hiki」,hikiwiki.org/ja/about.html,2016/9/2 アクセス.

[2]「Rubygems-Wikipedia」,ja.wikipedia.org/wiki/RubyGems,2016/9/2 アクセス.

[3]「optparse の使い方」,dharry.hatenablog.com/entry/20081008/1223490673,2016/9/2 アクセス.

[4]「パーフェクト Ruby」,Ruby サポーターズ (技術評論社 2013).