『PythonでRakeもどきを作ってみた』

makoto kuwata <kwa@kuwata-lab.com>

pyKook - Python版 Rake

* タスク定義をPythonで記述
* タスク処理を「料理」に例える
  * Recipe (タスク定義), Cookbook (定義ファイル), Product (生成物), Ingredient (材料), ...
* http://www.kuwata-lab.com/kook/pykook-users-guide.html

タスク定義

Kookbook.py

```
@recipe
def clean(c):
    '''remove *.o files'''
    rm_rf("**/*.o")
```

@recipe をつけると
タスク定義に

関数ドキュメントが
タスクの摘要に

コマンドでは
「**/」が利用可能

タスクの一覧と実行

タスクの実行

```
sh> kk clean
### * clean (recipe=clean)
$ rm -rf **/*.o
```

・kk または pykook で起動
・-l または -L で一覧表示

タスク一覧の表示

```
sh> kk -l
Properties:

Task recipes:
  clean        : remove *.o

File recipes:
```

ファイル生成ルール

Kookbook.py

```
@recipe
@product('*.o')
@ingreds('$(1).c', if_exists('$(1).h'))
def file_o(c):
    '''compile *.c into *.o'''
    system(c%"gcc -c $(ingred)")
```

@product() に
パターンを指定

もしファイルが
存在するなら

@ingreds() で
材料を指定

キーワードや変数を埋め込み可能
(例: $(ingred) は最初の材料を表す)

内容が変更されたか確認

タスク別オプション

Kookbook.py

@spices でタスクごとの command-line option を定義

commad-line の引数とオプションを受け取る

```
@recipe
@spices("-d:  debug mode",
        "-p port:  port number")
def start(c, *args, **kwargs):
    debug = kwargs.get('d')   # True or None
    port  = kwargs.get("p", 8080)
    app   = args and args[0] or 'helloworld'
    echo(c%"debug=$(debug), port=$(port), app=$(app)")
```

実行例: sh> kk start -dp 8080 helloworld

コマンドフレームワーク

mygit

```
#!/usr/bin/env pykook -X

kook_desc = "git helper"

@recipe
def ci(c):
    '''invokes git commit'''
    system("git commit -a")

@recipe
def co(c):
    '''invokes git clone'''
    system(c%"git clone $(url)")
```

「pykook -X」を shebang に指定

コマンドの説明

各タスクがサブコマンドに

実行例

サブコマンドの実行

```
sh> ./mygit ci
[master 773f3ba] comment
 1 files changed,
 1 insertions(+),
 0 deletions(-)
```

サブコマンド一覧

```
sh> ./mygit -h
mygit - git helper


sub-commands:
   ci       : invokes git commit
   co       : invokes git clone
```

・スクリプト名で起動する
　（kk や pykook ではないことに注意）
・サブコマンド別のオプションも可能

プロパティとプロパティファイル

Kookbook.py

```
CC = prop("CC", "gcc")
cflags = prop("cflags", "")
@recipe
def build(c):
    system(c%"$(CC) $(cflags) hello.c")
```

Properties.py

```
cflags = "-g -Wall"
```

コマンドラインでも
「--cflags='...'」で指定可能

定義済みコマンド

* cp(), cp_p(), cp_r(), cp_pr()
* rm(), rm_r(), rm_f(), rm_rf()
* mkdir(), mkdir_p()
* mv()
* store()　# ディレクトリ構造を保ったままコピー
* edit()　# ファイルを一括変更
* system()
* echo()
* touch()
* cd(), chdir()

```
with chdir('build'):
    system('...')
# withを抜けると
# 元の場所に自動復帰
```

その他の機能

* 「-n」でコマンドを実行せず表示のみ（dry-run）
* 「-F」で強制実行（タイムスタンプを無視）
* 「-D」でより詳細な情報を表示
* タスク定義のループを自動検出

続きはWebで！

http://www.kuwata-lab.com/kook/pykook-users-guide.html

thank you