Python での DB の扱い方（peewee編）

postgresql の導入 - Pythonでの接続テストまで

参考1 : ライブラリ：peewee
(http://www.lifewithpython.com/2014/05/python-peewee.html)
参考2 : peewee document Managing your Database (http://docs.peewee-orm.com/en/latest/peewee/database.html)
参考3 : Homebrewを使ったPostgreSQLのインストール
(http://qiita.com/tstomoki/items/0f1a930bd42a8e1fdaac)
参考4 : PostgreSQLの基本的なコマンド (http://qiita.com/H-A-L/items/fe8cb0e0ee0041ff3ceb)
参考5 : Welcome to Peewee, a lightweight python ORM
(http://nbviewer.jupyter.org/gist/coleifer/d3faf30bbff67ce5f70c)

brew で postgresql を導入

shell
```
# install
$ brew install postgresql
# DBの初期化
$ initdb /usr/local/var/postgres -E utf8
```

postgresql の起動

ひとつのシェルで postgres を起動しておく

shell-A
```
$ postgres -D /usr/local/var/postgres
```

もうひとつのシェルを立てて，以下のコマンドでデータベース一覧が取得できるか確認する.
一覧が取得できれば，インストール成功かつDBは正常に動作している.

shell-B
```
$ psql -l

    Name    |   Owner   | Encoding |   Collate   |    Ctype    |
Access privileges
```

```
-----------+----------+----------+-------------+-------------+------
------------------
 postgres   | Yoshiharu | UTF8     | ja_JP.UTF-8 | ja_JP.UTF-8 |
 template0  | Yoshiharu | UTF8     | ja_JP.UTF-8 | ja_JP.UTF-8 |
=c/Yoshiharu           +
            |           |          |             |             |
Yoshiharu=CTc/Yoshiharu
 template1  | Yoshiharu | UTF8     | ja_JP.UTF-8 | ja_JP.UTF-8 |
=c/Yoshiharu           +
            |           |          |             |             |
Yoshiharu=CTc/Yoshiharu
(3 rows)
```

postgresql へのログイン & DB作成

shell

```
# login
~$ psql -U Yoshiharu -d postgres

# login後は shell の表記が "postgres=#" になる
# test 用の DBを作成する
postgres=# create database peeweetest;
# 確認
postgres=# \l

  Name    |   Owner   | Encoding |   Collate   |    Ctype    |
Access privileges
-----------+----------+----------+-------------+-------------+------
------------------
peeweetest | Yoshiharu | UTF8     | ja_JP.UTF-8 | ja_JP.UTF-8 |
 postgres   | Yoshiharu | UTF8     | ja_JP.UTF-8 | ja_JP.UTF-8 |
```

python での postgresql の操作

作業用ディレクトリとpython仮想環境の作成

shell

```
$ mkdir peewee_test
$ cd peewee_test
# 仮想環境の作成
$ pyenv virtualenv 2.7.11 peewee_test
New python executable ...
Installing setuptools, pip, wheel...done.
# 仮想環境をローカルに適用する
$ pyenv local 2.7.11/envs/peewee_test
```

SQL操作用の package のインストール

```
$ pip install peewee psycopg2
```

インストールできているかを確認

shell
```
$ python
>> import peewee
>> print(peewee.__version__)
2.10.1
```

さきほど作成したテスト用DB(pythontest)にテーブルを追加してみる
だいたいの流れは以下のとおり.

1. peewee を import する
2. host名, DB名(peeweeTest), ユーザ名を指定して, DB接続用のドライバを生成する.
3. 定義した class を元に, table を作成する

今回は, localhost (=host_name) の peeweetest (=db_name) なので, url は以下のソースコードで接続完了する.

connectDB.py
```
from peewee import *

db = PostgresqlDatabase(
        'peeweetest',       # db_name
        user='Yoshiharu',   # user_name
        host='localhost' # host_name
        )
```

次に同じソースコード内でクラスを定義し, table を作成する.

connectDB.py
```
# define person class
class Person(Model):
    name = CharField()
    birthday = DateField()
    class Meta:
       database = db

    def __init__(self, name, birthday):
        super(Person, self).__init__()
        self.name = name
        self.birthday = birthday
```

実行後, postgres 内の peeweetest に person という table が生成されているこ

とを確認する.

```shell
$ pip install ipython # optional
$ ipython (or python)
# create table
In [1]: from connectDB import Person
In [2]: Person.create_table()

# connect Database
$ psql -U Yoshiharu -d peeweetest
# check tables
peeweetest=# \dt

          List of relations
 Schema |  Name  | Type  |   Owner
--------+--------+-------+-----------
 public | person | table | Yoshiharu
(1 row)
```

同じように，person table の中に 'Bob' という data を追加してみる

```shell
$ ipython

In [1]: from connectDB import Person
In [2]: from datetime import date
In [3]: Bob = Person(name='Bob', birthday=date(1960, 1, 15))
In [4]: Bob.save()
Out[4]: 1

# after DB peeweetest login
peeweetest=# select * from person;

 id | name |  birthday
----+------+------------
  1 | Bob  | 1960-01-15
(1 row)
```