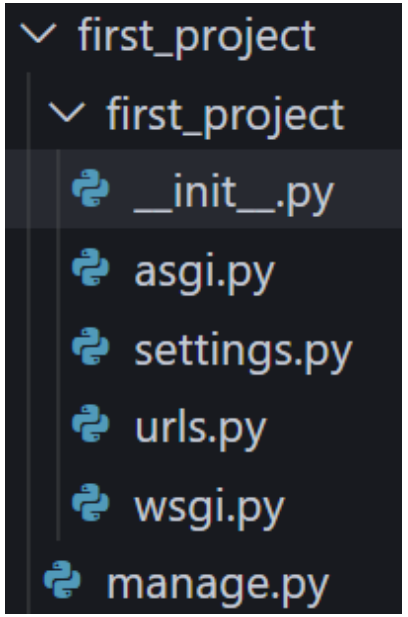
Django基本講座 1
(Django導入編)

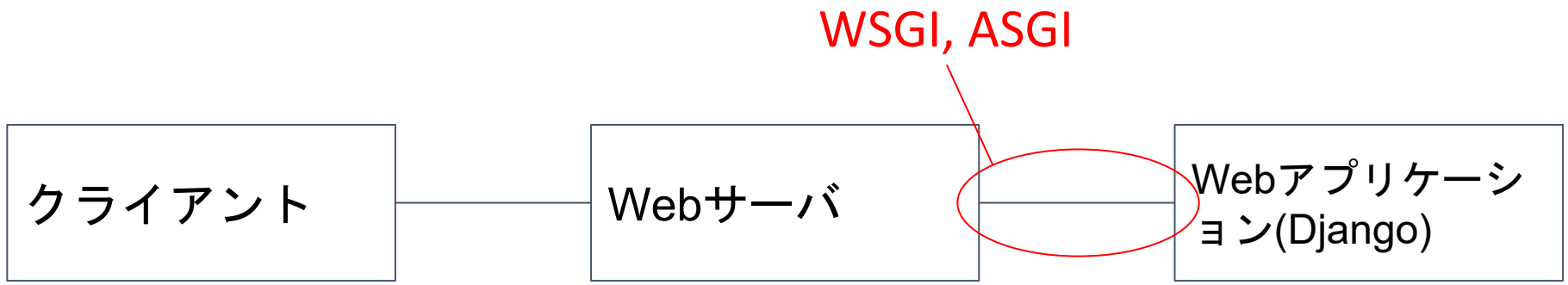
Djangoの各ファイルの役割

myproject/　・・・Djangoのプロジェクトの入ったフォルダ。
　manage.py・・・アプリケーション作成、マイグレーション等様々な操作を行うためのファイル
　myproject/・・・実際のプロジェクトのパッケージ。
　　__init__.py・・・このディレクトリがPythonのパッケージであることを知らせるためのファイル。中には何も記載しなくてよい
　　settings.py・・・Djangoプロジェクトの設定ファイル
　　urls.py・・・URLのディスパッチを設定する
　　wsgi.py・・・WSGI(Web Server Gateway Interface)としてデプロイする際に用いられる。
　　asgi.py(Django 3.0から)・・・ASGI（非同期サーバゲートウェイインタフェース）としてデプロイする際に用いられる



詳細: https://docs.djangoproject.com/ja/3.1/intro/tutorial01/

WSGI(ウィスギ)とASGI(アスギ)とは



- WSGIは、要求を受け取って、応答を返す**同期**呼び出し。
- 非同期に対応していない。WebSocket HTTP/2.0に非対応という課題

- ASGI(Asynchronous Server Gateway Interface)の略
- WSGIの精神的な後継者(a spiritual successor)
- HTTP/1.1, WebSocket, HTTP/2.0に対応
- 非同期での呼び出しが可能

Djangoプロジェクトの起動

以下のコマンドを実行すると、Djangoのプロジェクトを起動できる

```
python manage.py runserver
```

ポート番号を指定するには以下のようにして実行します(デフォルトは800です)

```
python manage.py runserver 8080　# 8080ポートで実行
```

IPアドレス1.2.3.4上でポート7000で待ち受ける場合

```
python manage.py runserver 1.2.3.4:7000
```

初期データベース、テーブル作成

settings.py内にプロジェクトの様々な設定が記載されています。
そのうち、DATABASESは、データベースの設定を記載したもので、
django.db.backends.sqlite3, django.db.backends.postgresql,
django.db.backends.mysql, django.dbbackends.oracleのいずれかに設定する

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.sqlite3',
        'NAME': BASE_DIR / 'db.sqlite3',
    }
}
```

ENGINE: 利用するデータベース
NAME: DATABASEの名前
sqlite以外の場合は、接続先のホスト名、パスワード、ユーザ名なども併せて記載が必要

参照先: https://docs.djangoproject.com/ja/3.1/ref/settings/

初期データベース、テーブル作成

以下コマンドを実行するとDjangoのプロジェクト用のテーブルが作成される

python manage.py migrate

auth_group・・・管理画面(後述)に存在するユーザのグループを格納する
auth_group_permissions・・・auth_groupとauth_permissionを結び付ける
auth_user・・・ログインユーザの情報を格納する
auth_user_groups・・・auth_userとauth_groupを結び付ける
auth_user_user_permissions・・・auth_userとauth_permissionを結び付ける
auth_permission・・・権限情報を格納する
django_admin_log・・・ログ情報を格納する
django_content_type・・・アプリケーションとアプリケーション内のModelの情報を格納している
django_migrations・・・マイグレーション(テーブル自動生成)の情報を管理する
django_session・・・セッションデータを格納する

アプリケーションの作成(startapp)

以下のコマンドを実行すると、Djangoのプロジェクト内にアプリケーションを作成できる

```
python manage.py startapp firstapp
```

以下のファイルが自動的に作成される

firstapp/
- __init__.py
- admin.py・・・管理画面
- apps.py・・・アプリケーションに関する設定
- migrations/・・・マイグレーションの設定内容が記載される
  - __init__.py
- models.py・・・モデルを作成
- tests.py・・・テストコードを作成
- views.py ・・・ビューを作成

作成したアプリケーションをsettings.pyに追加する

これをしないとDjangoがアプリケーションを認識しない

```
INSTALLED_APPS = [
  my_app
]
```

最初のViewの作成

アプリケーション内のビューに以下の内容を追加

```
from django.http import HttpResponse

def index(request):
    return HttpResponse(‘<h1>Something</h1>’’) # リクエストが来たらこの値をレスポンスとして返す
```

アプリケーションのフォルダに**urls.py**を作成し、以下の内容を記述（**URLディスパッチ**）

```
from django.urls import path

from . import views

app_name = ‘first_app’ # appの名前空間を表す（画面の遷移先を指定する場合に用いる）
urlpatterns = [
    path('', views.index, name='index') # /でアクセスした場合にviewsファイル内のindex関数を返す
]
```

プロジェクトのフォルダのurls.pyに、アプリケーションフォルダ内のurls.pyのパスを通す

```
path(‘app/’, include(‘app.urls’))
```

Viewを複数作成してurlディスパッチする

views.pyに**複数の関数を定義**して、urls.pyでその定義先を設定してurlディスパッチの設定をする

```
urlpatterns = [
  path(1つ目のpath),
  path(2つ目のpath),
]
```

urlに引数を取って、関数に渡すこともできる
例えば、
path('page/<str:name>', views.page, name='page')
とした場合、<str:name>の部分が引数となって
関数内で利用することができる。

```
def page(request, name):
    # nameを利用した処理
```

<**str**:name>
としているのでstr型で変数を渡す

str	/を含まない文字列型、何も指定しない場合はデフォルトでstrになる
int	数値型、0または正の整数しか受け付けない
slug	英数字、-, _から成る文字列 例) `building-your-1st-django-site`
uuid	UUID。英数字の集まりとハイフンから成る `075194d3-6885-417e-a8a8-6c931e272f00`
path	/を含んだ文字列

問題

1. FirstExamというプロジェクトを作成しましょう
2. FirstAppというアプリケーションを作成しましょう
3. migrateを行って、Djangoのデフォルトのデータベースを作成しましょう
4. FirstAppの中に以下の画面を作成して、それぞれ各URLで遷移できるようにしましょう
   4-1. http://127.0.0.1:8000/first/add/num1/num2
   → num1とnum2の値を足したものを表示する(ただし、num1とnum2は必ず0か正の整数)
   4-2. http://127.0.0.1:8000/first/minus/num1/num2
   → num1とnum2の値を引いたものを表示する(num1とnum2は浮動小数点数で0かプラス)
   4-3. http://127.0.0.1:8000/first/div/num1/num2
   → num1とnum2の値を割ったもので四捨五入したものを表示する(num1とnum2は浮動小数点数で0かプラス)