Djangoの静的ファイルをNginxから配信する-お名前ドットコムVPSへ移行への道14

Djangoで静的ファイルを WhiteNoise による直接配信から、Nginx による配信へ移行しました。 この記事では、以下の点を丁寧に解説します。

- BASE_DIR が実際にどこを指しているのか
- 開発時の static と staticfiles の関係
- STATIC_URL / STATIC_ROOT / STATICFILES_STORAGE の役割
- STATIC_URL と実際の HTTPS パスの関係
- Docker + Nginx で静的ファイルを共有する方法
- 複数 Django プロジェクトで 1 つの volume を共有してよいか
- WhiteNoise と Nginx の違いとパフォーマンス比較

こちらが上記方法で配信してるアプリです。

settings.pyの各変数は実際にどこを指してるのか確認

まず、私の環境を例に、settings.py の各変数が実際のコンテナ内でどのパスに対応しているのか整理します。

ディレクトリ構成（抜粋）Dckerfile、docker-compose.yml、settings.py

ホストのディレクトリ構成

```
django2/
├── docker-compose.yml
├── Dckerfile
├── ac_web2ren/
│       ├── access_counter/
│       │       └── settings.py
│       ├── hajimete_app
│       │       └── static/
│       └── staticfiles/
│                └── hajime/
wafvps/
├── docker-compose.yml
├── modesecurity/
│       └── jikken1.conf
```

各ファイルの関係ある部分だけを抜粋してます。

```
Dckerfile
WORKDIR /code
COPY . /code/

django2/docker-compose.yml
services:
  # --- 各アプリケーションコンテナ ---
  ac_web:
    build: .
    volumes:
      - .:/code
    container_name: django_ac_web
    working_dir: /code/ac_web2ren
```

この設定により、ホスト側の django2/ がコンテナ内の /code と完全に共有されます。

```
django2/ac_web2ren/access_counter/settings.py

BASE_DIR = os.path.dirname(os.path.dirname(os.path.abspath(__file__)))

DEBUG =False

MIDDLEWARE = [
```

```
    'whitenoise.middleware.WhiteNoiseMiddleware',
]

STATIC_URL = "acweb2ren/static/"
STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')
STATICFILES_STORAGE =
"whitenoise.storage.CompressedManifestStaticFilesStorage"

print("static_rootは以下です。")
print(STATIC_ROOT)
```

BASE_DIRとSTATIC_ROOTの実際の値を確認

まず、実行してBASE_DIRとSTATIC_ROOTが実際に何が設定されているかをprintして調べてみます。
print(STATIC_ROOT)の結果です。
static_rootは以下です。
/code/ac_web2ren/staticfiles

STATIC_ROOT = os.path.join(BASE_DIR, 'staticfiles')によって、BASE_DIRと
statifilesをつなげたものなのでBASE_DIRは/code/ac_web2renということになります。

/codeは
Dckerfileで定義されてる
WORKDIR /code
さらにdocker-compose.ymlで以下のようにホストとバインドマウント（共有）されてる。

```
  volumes:
    - .:/code
```

ホストでの「.」の表す位置は、このvolumes:の記述があるdocker-compose.ymlの場所です。
また、これはバインドマウント（.から始まる場合はバインドマウント）で、ホストの.とコンテナの
codeが共通のディレクトリ、ファイルになり、どちらかにあるディレクトリ、ファイルを変更すると
もう片方に反映されます。
具体的には、上記のホストのディレクトリ構成のdjango2が、コンテナの/codeと共通になります。

コンテナの中に入って確認してみると

```
docker exec -it コンテナ名 bash
root@0046ef6baff3:/code# ls
Dockerfile     docker-compose.yml       requirements.txt     ac_web2ren    他
```

これはホストでdjango2でlsを実行したときと同じ結果です。

BASE_DIRの定義を解説

今度は、BASE_DIRを定義面から解説します。
BASE_DIRを定義してるsettings.pyがコンテナのどこにあるか確認する
と/code/ac_web2ren/access_counter/settings.pyにある。