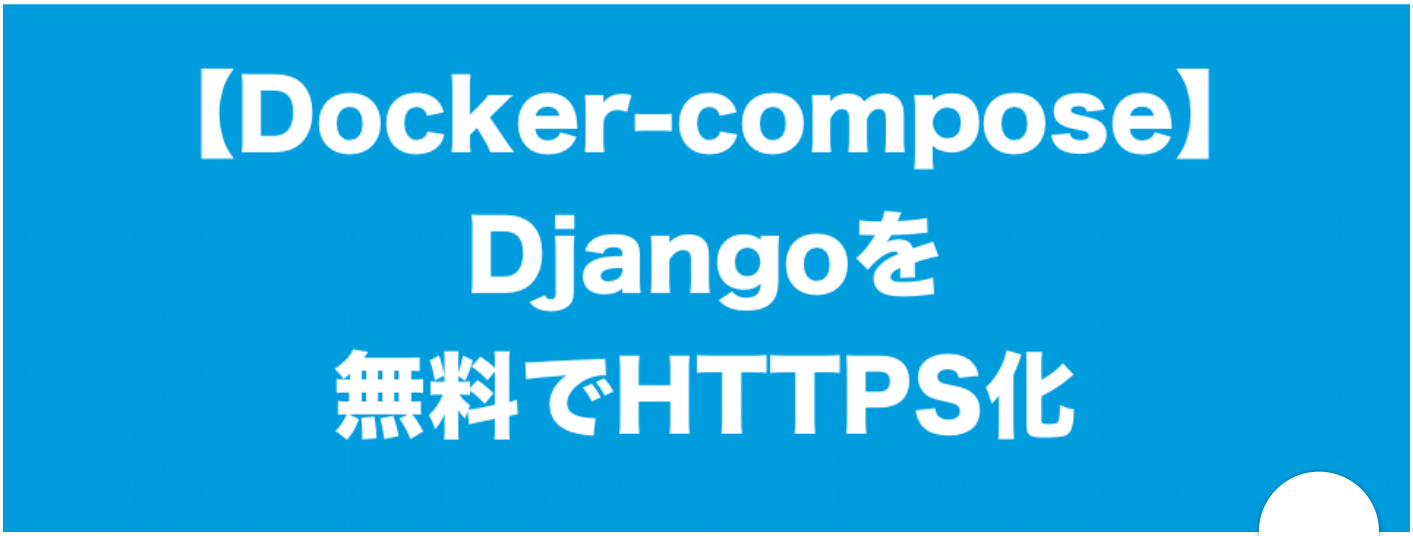
[Docker]Djangoを無料でHTTPS化して簡単にデプロイする方法

2018/06/12　2020/03/08



こんばんは、エンジニアの眠れない夜です。

Djangoを使っているエンジニアに朗報です！

このdocker-composeを使えばDjangoをHTTPS化して簡単にデプロイできるようになります。

しかも、HTTPS化が無料です！

こんな嬉しいことはありませんね＼(^o^)／

- サーバー環境を毎回設定してデプロイする。
- サーバーと開発環境との差分を吸収する。
- サーバー環境構築時間がほぼ0になる。

という素晴らしいメリットがあります。

私はサーバー環境を設定するのがものすごく嫌いです(；´Д｀) Linuxコマンドをイジイジするのがそもそもあまり好きでないのが大きな要因だと思います。

そして、プログラマーですが、インフラエンジニアではありません！

言いたいことてんこ盛りですが本題に入ります！（笑）

目次 [非表示]

Django×uWSGI×Nginxを一発で設定してくれるDockerFile

1ヶ月ほど前にDockerの存在を知りました。（おそ！Σ(ﾟ∀ﾟﾉ)ﾉ

Dockerを使えばサーバー環境を毎回構築しなくて良いことを知り、これはなんとしても利用したいと思いました。

そして、まずはDjangoを簡単にデプロイする方法がないか調べてこちらを見つけました。

https://hub.docker.com/r/dockerfiles/django-uwsgi-nginx

※ 以前はGithubがあったのですが現在では無くなっているのでDcokerHubのリンクに変更しました。

dockerfiles/django-uwsgi-nginxを利用すると名前のまんまですがdjango×uwsgi×nginxの設定を全て勝手にやってくれます。

最高に楽ちんです。幸せです！

Django-uWSGI-Nginxに足りないもの

しかし！一つだけ足りないものがあります。

ココ最近Googleが積極的にHTPPS化を進めたことで、今までは不要だったHTTPS化というものがドメインごとに必要になりました。

このHTTPS化というのが面倒くさくて、いちいち証明書を発行しなくてはいけません。

しかも、この証明書が年間800円〜4000円。サービスを作るたびにこんなの毎回払うのってすごく嫌ですよね。

let's encrypt という無料で証明書を発行してくれるサービスがあるのですが、certbotをインストールして証明書を発行して、90日？程度で再発行しないと証明書が無効になります。

これを使ったこともあるのですが、証明書を発行するまでインストールに始まりエラーを何度か吐いて、証明書を手に入れるまで... まぁ面倒くさいです。

サイトのHTTPS化を自動でやってくれるDockerコンテナ

そして次に出会ったのがこちらです。

https://github.com/SteveLTN/https-portal

これ作った人天才ですね。感動＆感謝です。これのおかげでどれだけ多くのエンジニアの命（時間）が救われることか...

SteveLTN/https-portal の使い方

ドメインとサーバーの接続確認

本番環境で利用する場合はDNS設定が完了し、ドメインがサーバーに向いていることを確認してください。

こちらのサイトでホスト名を入力して「実行」すると「入力の逆引き または 正引き」にサーバーのIPアドレスが表示されればOKです。

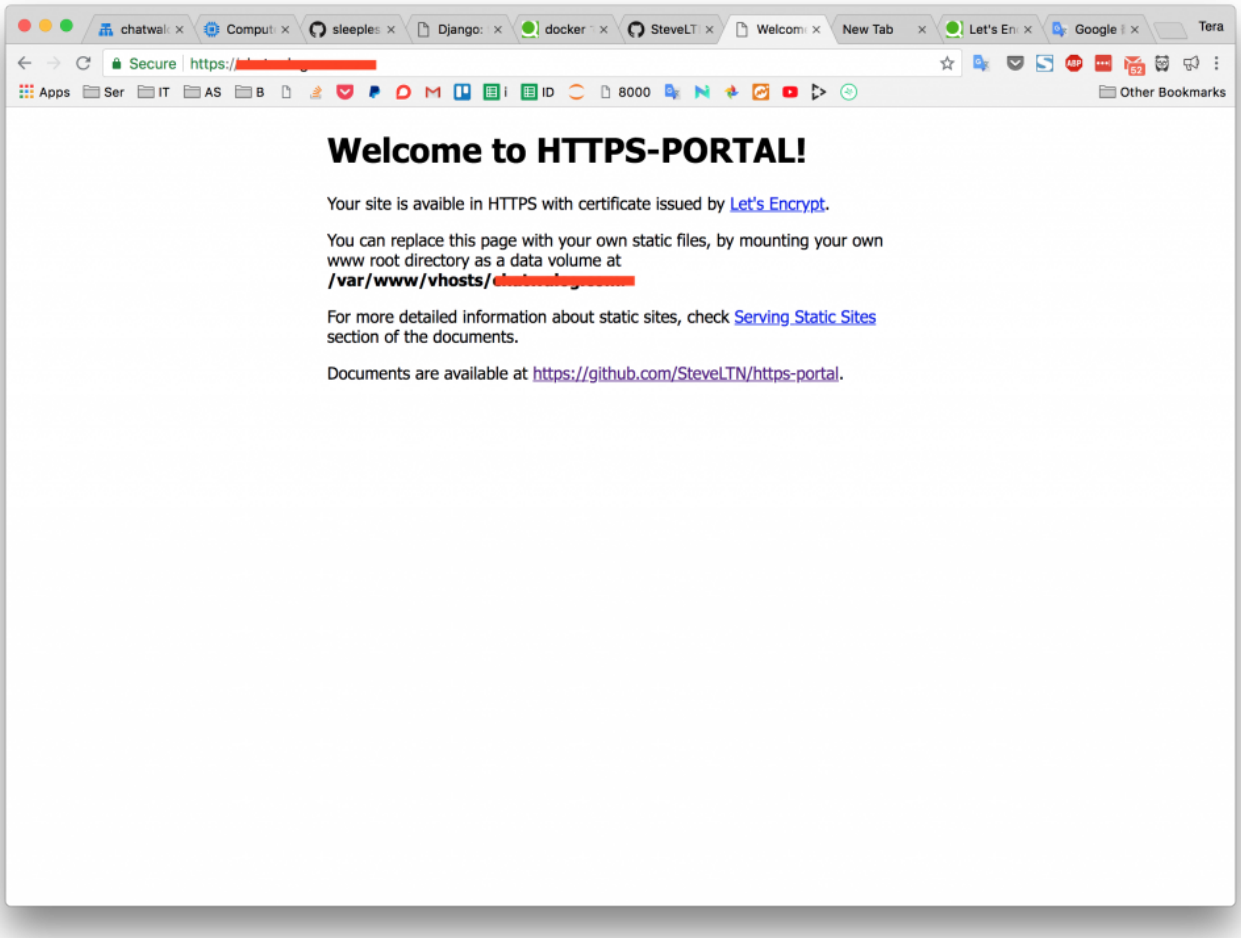
docker-compose.yml を作成・実行

サーバーで下記のように docker-compose.yml を作成します。

docker-compose up を実行しますこれだけで証明書が発行されます。

※ 署名が発行されるまでは時間がかかります。一度署名を作成すると次回以降はサクッと表示されます。

```
https-portal:
  image: steveltn/https-portal:1
  ports:
    - '80:80'
    - '443:443'
  environment:
    DOMAINS: 'example.com'
    STAGE: 'production'
```



素晴らしい！！HTTPS-PORTALさんナイスです！見事にHTTPS化されています(´ω｀)

STAGE: ‘production’ を STAGE: ‘local’ とすればオレオレ証明書を発行して開発環境でテストてきます。その時は **hosts** の設定にドメインを追加することをお忘れなく！

※ DOMAINS: のドメインは利用したいドメインに置き換えてください。

SteveLTN/https-portalについてもっと知りたい方は色んなオプションがあるので詳しくはこちらを参照

SteveLTN/https-portal と Django-uWSGI-Nginx を統合

ここで本題です。前置きが長かったですね。(；´∀｀)

DjangoとHttps-portalをどうやって繋げば良いんだろ？といろいろ試行錯誤した結果一番簡単な方法はこれだと思います。

docker-compose.yml

```
version: '3'
services:
  https-portal:
    image: steveltn/https-portal:1
    ports:
      - '80:80'
      - '443:443'
    environment:
      DOMAINS: 'example.com -> http://django:8080'
      STAGE: 'local' # or 'production'
    # volumes:
    #   - /var/lib/ssl_certs:/var/lib/https-portal

  django:
    build: ./django-uwsgi-nginx
    volumes:
      - ./django-uwsgi-nginx/app:/code/app
    ports:
      - '8080'
```

DockerComposeを使い慣れている人ならたったそれだけ！？と思うほど簡単ですね(^_^;)

慣れてないとこれだけのことでもなかなか時間を要しましたm(_ _)m

私のようなDocker初心者の為に先程の docker-compose.yml の説明を加えると

```
DOMAINS: 'example.com -> http://django:8080'
```

example.com のアクセスを django のポート 8080 に転送する設定です。

と、いうことはDocker内で先程の Django-uWSGI-Nginx を動かせばいいだけなので

```
django:
```

```
build: ./django-uwsgi-nginx
volumes:
    - ./django-uwsgi-nginx/app:/code/app
ports:
- '8080'
```

を実行します。 build: ./django-uwsgi-nginx は django-uwsgi-nginx 内のDockerFileを使ってね！という指示です。もちろんビルドした後のイメージを指定しても大丈夫です。

ポートはDockerコンテナ側で8080を指定します。

※ここでポートを80、443に設定するとhttps-portalと衝突します。

※build: をimage: に変えて開発中のコンテナを利用する時はポート番号に注意してください。

Dockerコンテナ内で作成されたDjangoのプロジェクトを下記のコマンドで ./django-uwsgi-nginx/app と共有する設定です。

```
volumes:
    - ./django-uwsgi-nginx/app:/code/app
```

こうすることでローカルで加えた変更がコンテナ内にも反映されます。

説明が少し前後してしまいますが発行された証明書はコンテナ作り直すたびに消えてしまうので、ホストのローカルに保存することで毎回証明書を発行しなくてよくなります。

```
# volumes:
#   - /var/lib/ssl_certs:/var/lib/https-portal
```

この部分のコメントを外して利用してください。https-portalで作成された証明書がホストの /var/lib/ssl_certs に保存されます。

/var/lib/ssl_certs が見つからない！っと怒られたら mkdir -p /var/lib/ssl_certs でディレクトリを作成してください。

実際にサクッとDjangoをHTTPSで動かしてみたい人へ

Githubにリポジトリを上げたのでこちらからCloneしてください。

```
git clone https://gitlab.com/sleepless-se/django-uwsgi-nginx-https.git
```

example.com でローカル環境でテストしてみます。hostsが設定されていないとエラーになるのでhostsを編集します。（Mac／Linux）

```
sudo vim /etc/hosts
```

```
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
127.0.0.1       example.com
```

一番下に`127.0.0.1　　example.com`を追加します。これでexample.comにアクセスした時にインターネット上のexample.comではなく、ローカルの127.0.0.1に飛びます。

※ vim（編集画面）を抜ける時はキーボードの escを押して :qw です。

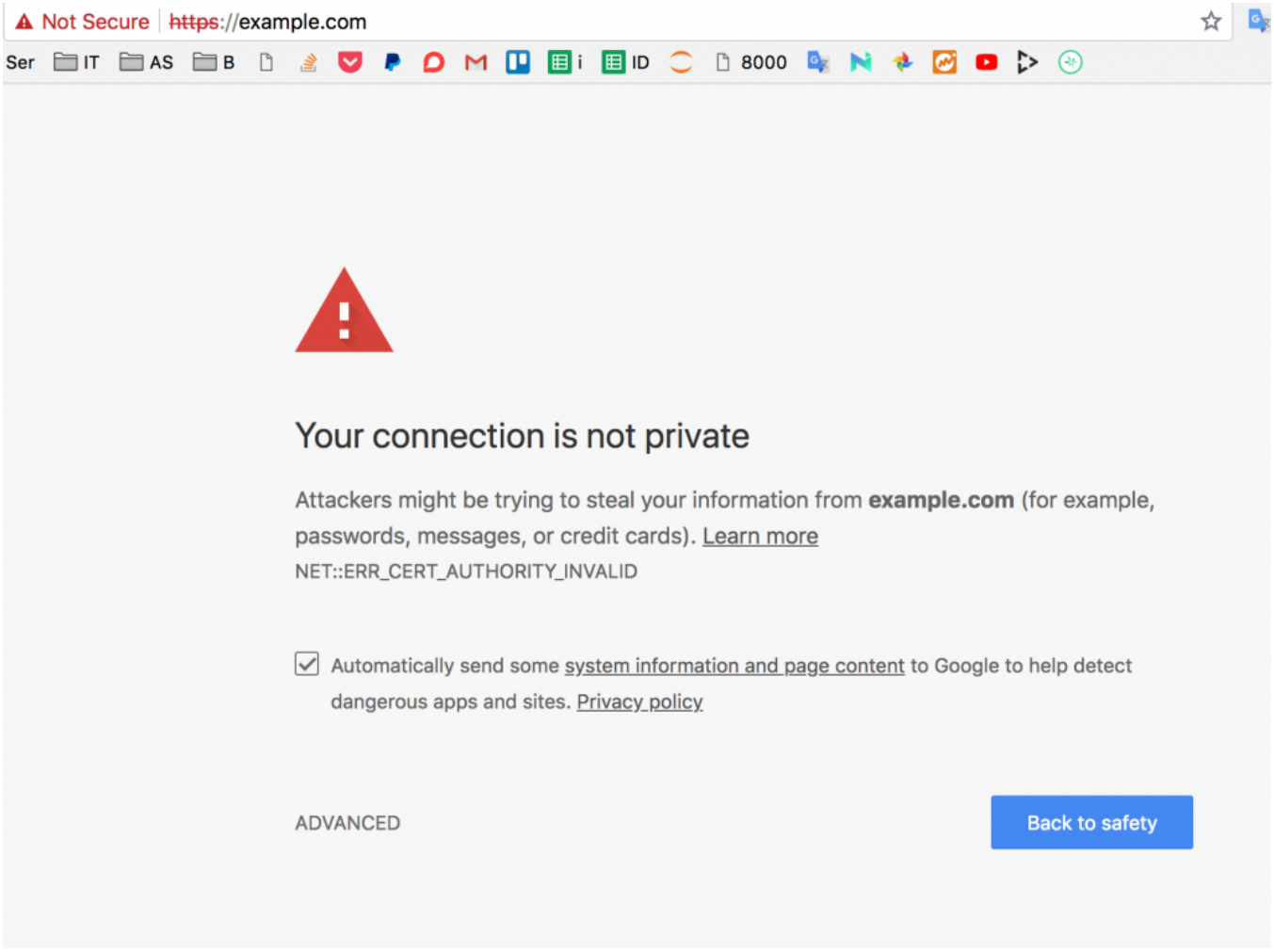
※ WindowsのHostsの設定方法

あとはdjango-uwsgi-nginxのフォルダの中に移動してdocker-compose upします。

```
cd django-uwsgi-nginx-https
```

```
docker-compose up
```

ザーッと文字が出てきたら　https://example.com/　を開きます。
※ 起動までしばらく時間がかかります。



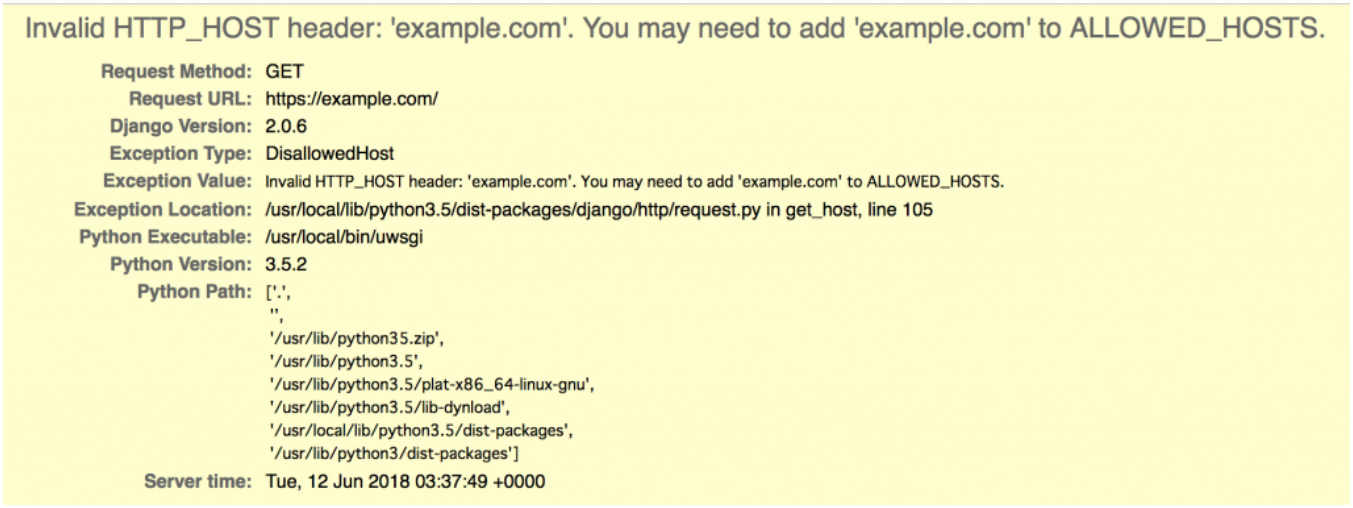
こんな画面が表示されれば成功です。

え？なんかエラーが出てるって？それは STAGE: 'local' でオレオレ署名を使うとこういうことになるんです。本番のサーバーの中で STAGE: 'production' にして、DNSが設定されているドメインならちゃんと表示されます。

この状態で確認するには画面左下の「ADVANCED」をクリックして、「Proceed to example.com (unsafe)」をクリックします。

すると 'example.com' が ALLOWED_HOSTS.に登録されていないよ！ってDjangoに怒られるので、ここからはDjangoで開発されているみなさんならもう大丈夫ですよね^^

setting.py の ALLOWED_HOSTS に examaple.com を加えればエラーが解決します。



ここで動いているDjangoは自分のプロジェクトでは無いのでどうやって置き換えればいいの？書き換えればいいの？という疑問が湧いてきます。

その答えは3ステップで解決します。

１．django-uwsgi-nginx/Dockerfile の下記のプロジェクトを作成する部分をコメントアウトます。

```
RUN django-admin.py startproject website /code/app/
```

２． django-uwsgi-nginx/app の中にあなたのDjangoプロジェクトをコピーします。

```
app
├poll
├mysite
├requirements.txt
└manage.py
```

↑例えばこんな感じになります。

３．wsgi.pyのパスをdjango-uwsgi-nginx/uwsgi.iniに設定します。

uwsgi.iniの中に module=website.wsgi:application と書いているところがあります。

Djangoプロジェクトのsettings.pyが入っているフォルダ名とwebappの部分を置き換えます。
例えば下記のようになります。

```
module=mysite.wsgi:application
```

この設定を忘れるとInternal Server Errorと言われます。

設定が終わったらビルドし直します。

ビルドし直さないと変更前のコンテナがキャッシュとして残っているのでそっちが起動して、「変更が適応されてない！？」ってなります。

```
docker build ./django-uwsgi-nginx/ -t django-uwsgi-nginx_django
```

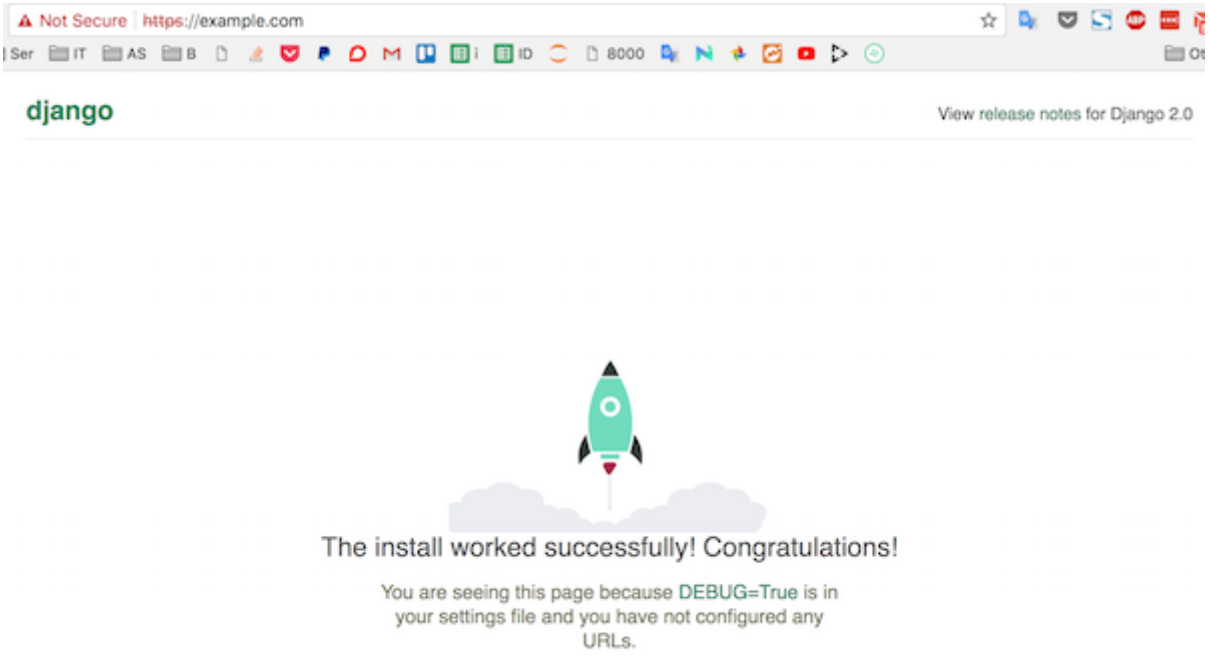
でビルドし直します。`build ./django-uwsgi-nginx/`はDockerFileがあるディレクトリのパスを指定

-t django-uwsgi-nginx_django でコンテナに `django-uwsgi-nginx_django` というタグを付けます。このタグが付いているコンテナが docker-compose 時に起動します。

```
docker-compose build --no-cache
```

こちらでもbuildをし直すことは可能ですが、全部ビルドし直すのでめっちゃ時間がかかります。

最後は docker-compose up で起動してhttps://example.com/に自分のプロジェクトが表示されれば完成です！



ミッションコンプリート！

最後に

Dockerの使い方っていろいろサイトを読んで勉強したんですけど、省かれている部分があって素人には分かりづらいものが多い印象です。

Docker初心者で別にDockerなんか勉強しなくていいからとにかく自分のDjangoプロジェクトをデプロイしたい！という方向けに説明をしたつもりです。

それでも、足りない部分があると思うのでうまくいかない人は気軽コメントください。

このページ見たらDjangoデプロイしたい人は誰でも簡単にデプロイできます！くらいの記事にしたいです。

コメントお待ちしております＼(^o^)／
追記：staticフォルダーのためのnginx-app.conf設定

自分で使っていてハマってしまったので追記します。自分のプロジェクトを django-uwsgi-nginx/app 以下に作成した時にそのままだとstaticフォルダをNginxから参照ができないのでnginx-app.confを編集する必要があります。

```
location /static {
    alias /code/app/プロジェクト名/static;
}
```

location /static の中をこの様に編集してください。メディアフォルダを利用するときも同様にnginx-app.confの編集が必要です。