uWSGIをCentos 7上で動かす

前回の続きで、今回はPythonのアプリケーションサーバ、uWSGIをCentos7で動かしてみましょう。

Linuxの入門知識が不安な方は、以下にまとめていますので適宜ご参照ください。
Linux学習講座 超入門編

Contents[hide]

概要

Centos上でuwsgiを動かしてみます。利用する環境は以下のとおりです。

Centos 7
Python 3.6
uWSGI(2.0.17.1)
Flask(1.0.2)

Webアプリケーションは/opt配下に配置します。また、Pythonの仮想環境として、venvを使用します。venvの環境も/opt配下に配置します。

```
/opt
├── apps
│   └── current
│       └── sample_api
└── envs
    └── sample_api
```

学習用としてPythonモジュールを/opt/apps/currentの下にflaskファイル直接配置していますが、実運用ではリリース用ディレクトリからシンボリックリンクを張るのが一般的です。それ以外はほぼそのまま本番で使える内容になっていると思います。

また、pidファイルは/var/run/uwsgi/uwsgi.pid、ログは/var/log/uwsgi/uwsgi.logに配置するものとします。

Python3インストール

Centos7が起動済みであることを前提とします。まずはPython3をインストールします。

```
sudo yum install -y https://centos7.iuscommunity.org/ius-release.rpm
sudo yum install -y python36u python36u-libs python36u-devel python36u-pip
```

次に、アプリケーション用フォルダの作成と、アプリケーション実行用のユーザーを作成します。

```
# アプリケーション配置用
sudo mkdir -p /opt/apps/current

# venv配置用
sudo mkdir -p /opt/envs

# ログ配置用
sudo mkdir -p /var/log/uwsgi

# pidファイル配置用
sudo mkdir -p /var/run/uwsgi

# 実効ユーザー作成、権限設定
sudo useradd pyuser
sudo chown -R pyuser:pyuser /opt/apps/
sudo chown -R pyuser:pyuser /opt/envs/
sudo chown -R pyuser:pyuser /var/log/uwsgi
sudo chown -R pyuser:pyuser /var/run/uwsgi
```

環境設定とflaskWebアプリケーションの配置

次に、環境設定とWebアプリケーションを配置します。まずはvenvで環境を作成し、必要なモジュールをインストールしましょう。

```
# 実効ユーザーに切り替え
sudo su pyuser

# 実行用の環境を作成
cd /opt/envs/
python3.6 -m venv sample_api

# 環境切り替え
source /opt/envs/sample_api/bin/activate

# 必要なモジュールをインストールする
pip install uWSGI
pip install flask
```

この例では実効ユーザーを設定してvenvを使用していますが、docker等のコンテナで単一アプリケーションを動かす場合はrootでvenvを使わずに直接実行するほうが一般的と思います。

次に、アプリケーションを配置します。run.pyという名前のpythonファイルを配置します。

```
cd /opt/apps/current
mkdir sample_api
vi /opt/apps/current/sample_api/run.py
```

内容は前回使用した、適当なjsonを返すだけのプログラムです。

```
# run.py
from flask import Flask, jsonify

app = Flask(__name__)


@app.route('/')
def api_sample():
    """
    APIサンプル
    :return:
    """
    result = {"code": "001", "name": "apple"}
    return jsonify(ResultSet=result)


if __name__ == '__main__':
    app.run()
```

ここでvenvの環境が正しく動作しているかを確認するために、pythonコマンドの動作確認を行います。

```
cd /opt/apps/current/sample_api/
python run.py
```

http://localhost:8080にアクセスして値が返るかどうかを確認しましょう。

uwsgiの設定

uwsgiの設定を行います。uwsgi.iniを作成します。

```
vi /opt/apps/current/sample_api/uwsgi.ini
```

以下の通り記述します。

```
[uwsgi]
# wsgiの設定
current_release = /opt/apps/current/sample_api
chdir = %(current_release)
wsgi-file=%(current_release)/run.py
callable = app
master = True

# アクセス許可ホスト:ポート
http=0.0.0.0:8080

# ログ
daemonize = /var/log/uwsgi/uwsgi.log

# 実効ユーザー/グループ
uid = pyuser
gid = pyuser

# pidファイル
pidfile = /var/run/uwsgi/uwsgi.pid

# pidファイルクリア
vacuum = true

# プロセス数など
processes = 2
threads = 2
thunder-lock = true
max-requests = 3000
max-requests-delta = 300
```

今度はuwsgiの動作を確認します。以下のコマンドで外部からアクセスできることを確認してください。

```
uwsgi --http=0.0.0.0:8080 --wsgi-file=run.py  --callable app
```

uwsgiが問題なく動けばpyuserでの作業は一旦終わりです。ctrl + cでuwsgiを停止した後、exitで一旦抜けてください。

```
exit
```

サービス登録

最後にユニット定義ファイルを作成してサービス登録します。以下のコマンドでユニット定義ファイルを新規作成します。

```
sudo vi /etc/systemd/system/sample.service
```

記述内容は以下の通りとなります。

```
[Unit]
Description = sample daemon
After=network.target

[Service]
User=pyuser
Group=pyuser
WorkingDirectory=/opt/apps/current/sample_api
Environment="PATH=/opt/envs/sample_api/bin"
ExecStart = /opt/envs/sample_api/bin/uwsgi --ini uwsgi.ini
ExecStop = /opt/envs/sample_api/bin/uwsgi --stop /var/run/uwsgi/uwsgi.pid
Type = forking
RemainAfterExit = yes

[Install]
WantedBy = multi-user.target
```

systemctlをリロードして完了です。

```
sudo systemctl daemon-reload
```

サービスが開始/終了できることを確認してください。

```
# サービスの開始
sudo systemctl start sample

# サービスの終了
sudo systemctl stop sample
```