Установка и настройка Django

Последнее обновление: 05.12.2023

Перед началом работы с Django нам естественно надо установить интерпретатор Python. Подоробнее об этом можно почитать здесь.

Существуют разные способы установки Django. Рассмотрим рекомендуемый способ.

Пакетный менеджер pip

Пакеты Django размещаются в центральном репозитории для большинства пакетов Python - Package Index (PyPI). И для установки из этого репозитория нам потребуется пакетный менеджер **pip**. Менеджер pip позволяет загружать пакеты и управлять ими. Обычно при установке python также устанавливается и менеджер pip. В этом случае мы можем проверить версию менеджера, выполнив в командной строке/терминале команду **pip -V** (V - с заглавной буквы):

```
C:\Users\eugen>pip -V
pip 23.3.1 from C:\Users\eugen\AppData\Local\Programs\Python\Python312\Lib\site-packages\pip (python 3.12)

C:\Users\eugen>
```

Если pip не установлен, то мы увидим ошибку типа

```
"pip" не является внутренней или внешней командой, исполняемой программой или пакетным файлом
```

В этом случае нам надо установить pip. Для этого можно выполнить в командной строке/консоли следующую команду:

```
python -m ensurepip --upgrade
```

Если pip ранее уже был установлен, то можно его обновить с помощью команды

```
python -m pip install --upgrade pip
```

Установка виртуальной среды

Виртуальная среда или **venv** не является неотъемлимой частью разработки на Django. Однако ее рекомендуется использовать, так как она позволяет создать множество виртуальных сред Python на одной операционной системе. Благодаря виртуальной среде приложение может запускаться независимо от других приложений на Python.

В принципе можно запускать приложения на Django и без виртуальной среды. В этом случае все пакеты Django устанавливаются глобально. Однако что если после создания первого приложения выйдет новая версия Django? Если мы захотим использовать для второго проекта новую версию Django, то из-за глобальной установки пакетов придется обновлять первый проект, который использует старую версию. Это потребует некоторой дополнительной работы по обновлению, так как не всегда соблюдается обратная совместимость между пакетами. Если мы решим использовать для второго проекта старую версию, то мы лишиемся потенциальных преимуществ новой версии. И использование виртуальной среды как раз позволяет разграничить пакеты для каждого проекта.

Для работы с виртуальной средой в python применяется встроенный модуль **venv**

Итак, создадим вируальную среду. Вначале определим каталог для проектов django. Например, пусть это будет каталог **C:\django**. Прежде всего перейдем в терминале/командной строке в этот каталог с помощью команды cd.

```
cd C:\django
```

Затем для создания виртуальной среды выполним следующую команду:

```
python -m venv .venv
```

Модулю venv передается название среды, которая в данном случае будет называться ".venv". Для наименования виртуальных сред нет каких-то определенных условностей. Пример консольного вывода:

```
C:\Users\eugen>cd C:\django  Переход к папке будущей виртуальной среды

C:\django>python -m venv .venv  Создание виртуальной среды

C:\django>
```

После этого в текущей папке (C:\django) будет создан подкаталог ".venv".

Активация виртуальной среды

Для использования виртуальную среду надо активировать. И каждый раз, когда мы будем работать с проектом Django, связанную с ним виртуальную среду **надо активировать**. Например, активируем выше созданную среду, которая располагается в текущем каталоге в папке .venv. Процесс активации немного отличается в зависимости от операционной системы и от того, какие инструменты применяются. Так, в Windows можно использовать командную строку и PowerShell, но между ними есть отличия.

Активация в Windows в коммандной строке

Если наша ОС - Windows, то в папке **.venv/Scripts/** мы можем найти файл **activate.bat**), который активирует виртуальную среду. Так, в Windows активация виртуальной среды **в коммандной строке** будет выглядеть таким образом:

```
.venv\Scripts\activate.bat
```

Активация в Windows в PowerShell

Также при работе на Windows в папке **.venv/Scripts/** мы можем найти файл**activate.ps1**, который также активирует виртуальную среду, но применяется только в PowerShell. Но при работе с **PowerShell** следует учитывать, что по умолчанию в этой оболочке запрещено применять скрипты. Поэтому перед активацией среды необходимо установить разрешения для текущего пользователя. Поэтому для активации виртуальной среды в PowerShell необходимо выполнить две следующих команды:

```
Set-ExecutionPolicy -ExecutionPolicy RemoteSigned -Scope CurrentUser
.venv\Scripts\Activate.ps1
```

Активация в Linux и MacOS

Для Linux и MacOS активация будет производиться с помощью следующей команды:

```
source .venv/bin/activate
```

Далее я буду приводить примеры на основе командной строки Windows, однако все остальные примеры не будут зависеть от того, что используется - PowerShell или командная строка, Windows, Linux или MacOS. В любом случае после успешной активации слева от текущего каталога мы увидим в скобках название виртуальной среды:

```
C:\Users\eugen>cd C:\django

C:\django>python -m venv .venv

C:\django>.venv\Scripts\activate.bat  Активация виртуальной среды

(.venv) C:\django>  Виртуальная среда активирована
```

Установка Django

После активации виртуальной среды для установки Django выполним в консоли следующую команду

```
python -m pip install Django
```

Она устанавливает последнюю версию Django.

```
(.venv) C:\django>python -m pip install Django
Collecting Django
  Using cached Django-4.1-py3-none-any.whl (8.1 MB)
Collecting sqlparse>=0.2.2
  Using cached sqlparse-0.4.2-py3-none-any.whl (42 kB)
Collecting tzdata
  Using cached tzdata-2022.1-py2.py3-none-any.whl (339 kB)
Collecting asgiref<4,>=3.5.2
  Using cached asgiref-3.5.2-py3-none-any.whl (22 kB)
Installing collected packages: tzdata, sqlparse, asgiref, Django
Successfully installed Django-4.1 asgiref-3.5.2 sqlparse-0.4.2 tzdata-2022.1

(.venv) C:\django>
```

Если нам интересует конкретная версия Django, то мы можем указать ее при установке:

```
python -m pip install django~=4.0.0
```

Проверка установки

Чтобы убедиться, что все установлено правильно, мы можем перейти к интерпретатору python. Для этого введем в терминале команду

```
python
```

И затем выполним последовательно следующие две инструкции:

```
>>> import django
>>> print(django.get_version())
```

Консольный вывод в моем случае:

```
(.venv) C:\django>python
Python 3.10.1 (tags/v3.10.1:2cd268a, Dec  6 2021, 19:10:37) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import django
>>> print(django.get_version())
4.1
>>>
```

Деактивация виртуальной среды

После окончания работы с виртуальной средой мы можем ее деактивировать с помощью команды:

```
deactivate
```