Дисципліна «Об’єктно-орієнтоване програмування на Python»

Встановлення та налаштування Django

Медведєв Д.Г.

Налаштування Django відбувається через командний рядок.

Командний рядок є потужним інструментом, що дозволяє виконувати всі операції через текст. Розробники використовують її повсюдно.

На Mac системах командний рядок можна відкрити через *Terminal* , що знаходиться в `/Applications/Utilities`. Відкрийте нове вікно *Finder* , потім директорію *Applications* , в нижній частині буде папка *Utilities* , з якої подвійним натисканням клавіші можна запустити програму *Terminal* .

Windows має дві вбудовані командні оболонки: *Command shell* і *PowerShell* .Рекомендовано використовувати PowerShell, яка дещо потужніша.

Віртуальні оточення є незамінними аспектами програмування на Python. Це **ізольовані контейнери** , які складаються з необхідних для проекту програмних інструментів, що розглядається. Вони дуже важливі, оскільки за замовчуванням Python і Django встановлюються в одну і ту ж директорію. З цієї причини можуть виникнути проблеми, коли користувач хоче працювати з кількома проектами на одному комп'ютері. Що, якщо `ProjectA` використовує **Django 4.0** , а торішній `ProjectB` все ще на **Django 3.**?

Для управління віртуальними оточеннями використовується Pipenv . Pipenv схожий на `npm` і `yarn`з JavaScript/Node екосистем: він створює `Pipfile`, в якому знаходяться необхідні програмні інструменти, а також `Pipfile.lock` для забезпечення детермінованих збірок. "Детермінування" передбачає, що при кожному завантаженні програми в нове віртуальне оточення конфігурація залишатиметься незмінною.

Для того, щоб оцінити Pipenv в дії, створюємо нову директорію та встановимо Django.

```
pipenv install django
```

Зазирнувши в середину папки, бачимо, що в ній з'явилося два нових файли: `Pipfile` і `Pipfile.lock`. Тепер у нас є вся інформація, необхідна для створення нового віртуального оточення, проте поки що нічого не активовано. Щоб його активувати, необхідно виконати

`pipenv shell`.

Тепер створюємо новий проект Django з назвою test_project за допомогою наступної команди. Не забудьте в кінці поставити крапку .

```
django-admin startproject test_project .
```

Залишилося переконатись, що все працює. Для цього запустимо локальний веб-сервер Django.

```
python manage.py runserver
```

Тепер необхідно перейти за посиланням http://127.0.0.1:8000/, де відкриється початкова сторінка «привітання з Django.

Проте, в повному результаті виводу буде вказана додаткова інформація, серед якої буде попередження про незастосовані міграції `17 unapplied migrations`. Технічно, це попередження на поточному етапі ні на що не впливає. Django повідомляє про те, що ми ще не мігрували або не конфігурували вихідну базу даних. Поки база даних використовуватись не буде , тому попередження не вплине на кінцевий результат. Проте, треба виконати команду

```
python manage.py migrate.
```

щоб Django здійснив міграцію вбудованих програм. Тепер при повторному виконанні `python manage.py runserver` командний рядок виведе чистий результат

Для зупинки локального сервера використовується комбінацію `CTRL+C`. Після цього вийти з віртуального оточення необхідно за допомогою команди `exit`.