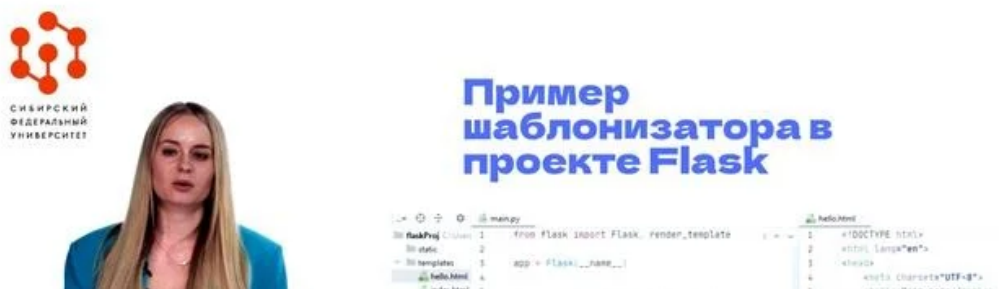
Разработка программного обеспечения на языке Python

Лекция 1. Концепции web-приложения

Посмотрите видеоуроки и ответьте на контрольные вопросы после лекции

Flask



00:00 / 05:41

В данной теме рассмотрим краткий обзор библиотеки flask для создания веб-приложений на python.

Для веб-разработки на Python используются специальные фреймворки - библиотеки, среди которых можно отметить django, flask, tornado, fastapi, bottle. В курсе далее будем рассматривать django и flask, так как они достаточно распространены и относительно просты в изучении.

Фреймворк **Flask** относится к микрофреймворкам.

Он является минималистичным каркасом веб-приложений, предоставляющим только самые базовые наборы инструментов. Это означает, что изначально при создании проекта на Flask не требуется делать много настроек – код выглядит очень простым и понятным. Дальнейшее расширение функциональности в проекте происходит за счет подключения библиотек. При этом, разработчику предоставляется возможность самостоятельно выбирать компоненты, которые он хочет использовать в своем проекте.

Давайте посмотрим, как выглядит минимальное приложение на Flask.

```python
from flask import Flask

app = Flask(__name__)


@app.route('/')
def hello():
    return 'Hello, World!'


app.run('localhost')
```

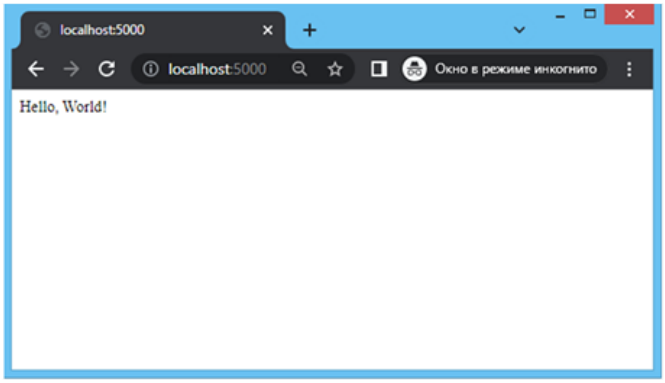
```
 * Serving Flask app '1' (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://localhost:5000 (Press CTRL+C to quit)
127.0.0.1 - - [25/May/2022 09:28:38] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [25/May/2022 09:28:38] "GET /favicon.ico HTTP/1.1" 404 -
```

В самом начале вам нужно подключить фреймворк, используя import, и создать объект приложения.

Затем создать функцию, к которой привязать URL (в данном случае это слэш, то есть корневой каталог сайта), и затем запустить приложение.

После запуска скрипта в консоли будет выведена ссылка с адресом нашего сайта.

Перейдя по нему, мы видим, что нам отобразилась строка “Hello world”, которую мы вернули в функции.



В проектах на Flask есть определенная **структура**.

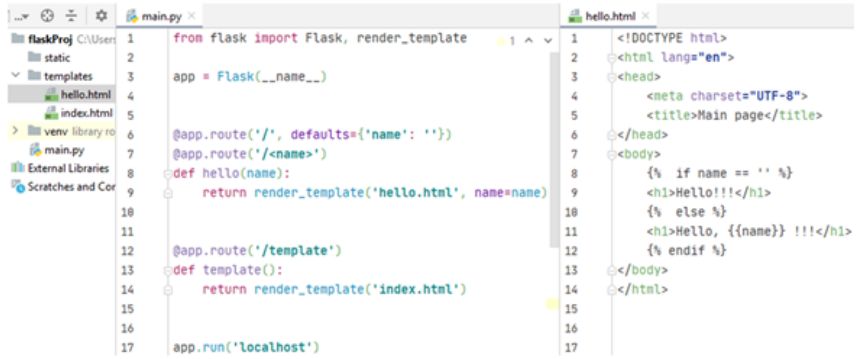
В некотором файле .py описаны маршруты и функционал сайта (у нас это main.py).

Кроме того, в проект могут быть добавлены директории static и templates. Static будет по умолчанию использоваться для хранения файлов стилей, скриптов и прочего, а templates – для страниц html.

Далее добавим указанные папки в наш проект, создадим в папке templates .html файл и добавим функцию, которая будет возвращать этот файл.

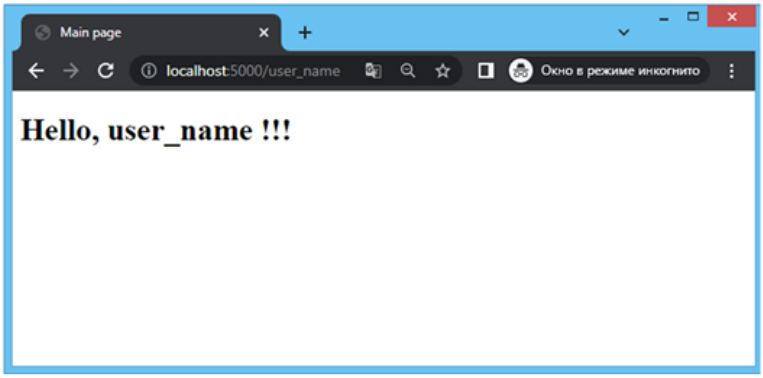
В комплекте с Flask идут следующие компоненты: Шаблонизатор Jinja2 (джинджа 2), который позволяет вносить определенную логику при генерации html-страниц. И набор инструментов Werkzeug (читается "Веркцойг") – набор инструментов стандартного интерфейса Python для развертывания веб-приложений и взаимодействия между ними и различными серверами.

Рассмотрим следующий пример.

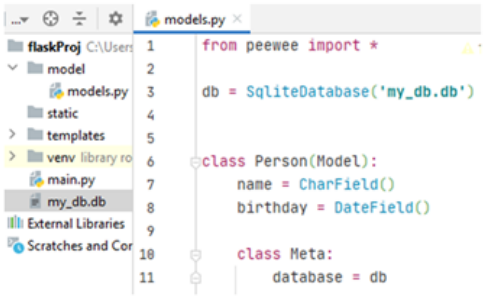


Немного изменим нашу функцию для начальной страницы, чтобы говорить hello не всему миру, а, допустим, какому-то пользователю. В декораторе функции укажем параметр name, значение по умолчанию его будет пустой строкой. Этот параметр мы передадим нашей новой html-странице, на которой кодом, похожим на Python, написано следующее: если строка пустая, выводим “Hello!”, если непустая – “Hello” и значение name. Этот искусственный пример демонстрирует гибкость, вносимую в генерацию страниц при использовании шаблонизатора. В его функционал входят не только условия, но и, например, циклы for. Подробнее про шаблонизатор будем говорить в следующих темах курса.

Так выглядит теперь наш проект, когда мы добавили еще одну ссылку, по которой возвращается html страница.



Давайте добавим в наш проект код для работы с базой данных. Для создания и работы с таблицами в python существует достаточно большое число различных библиотек. Создадим одну таблицу, которая будет хранить список людей. В качестве базы данных будем использовать sqlite.



Говоря про Flask как микрофреймворк, мы говорили о его масштабируемости и о том, что разработчик сам может выбирать, какие библиотеки использовать. Flask считается подходящим веб-фреймворком для создания небольших легковесных веб-приложений. Легкий для понимания. Понять, как работать с фреймворком, сможет даже начинающий программист. Flask имеет простую структуру и интуитивно понятный синтаксис. Кроме того, в отличие от других фреймворков Flask позволяет программисту полностью контролировать процесс разработки.

Некоторые недостатки данного фреймворка проявляются при разработке более крупных веб-приложений. Среди них можно отметить более медленную разработку и более высокие затраты ресурсов на обслуживание сложных проектов. Это связано с тем, что для создания такого проекта требуются некоторые предварительные знания фреймворка. Асинхронность фреймворка дает преимущества в функциональности, но может быть сложной для начинающих разработчиков. Также в фреймворке flask отсутствуют встроенные модули для базы данных и ORM. При этом flask предлагает ограниченную поддержку и меньшее сообщество по сравнению с Django.

В данной теме мы рассмотрели микрофреймворк Flask и создали просто приложение, демонстрирующее всю легкость и удобство этого фреймворка. Более объемным фреймворком Python является Django, который будем изучать на следующих лекциях.

Вопросы

ПРЕДЫДУЩИЙ ЭЛЕМЕНТ КУРСА

◄ Запись синхронных занятий по курсу "Веб-программирование на Python"

Перейти на...

СЛЕДУЮЩИЙ ЭЛЕМЕНТ КУРСА

Задание 1. Дизайн страницы приложения ►