



Python Practice

Проектирование Web сервиса



Python Practice

Автор курса



Крементарь Ксения

Ведущий Python разработчик

Системный архитектор

в компании K-Solutions

Python Practice

После урока обязательно



Повторите этот урок в видео формате на
[ITVDN.com](http://itvdn.com)



Проверьте как Вы усвоили данный материал на
[TestProvider.com](http://testprovider.com)

Проектирование Web сервиса

Python Practice

На этом уроке

1. Вспомним, что такое web разработка и web сервис.
2. Узнаем, что такое web фреймворки, зачем они нужны.
3. Узнаем, какие web фреймворки существуют для создания проектов на Python.
4. Узнаем, как создать простейшее web приложение с помощью фреймворка Flask.
5. Напишем и запустим первое Hello world приложение на Flask.

Python Practice

Web разработка — создание web сервисов!

Web разработка, как мы уже говорили в самом начале нашего курса — это создание, разработка web приложений, web сервисов. Web-сервис — специальная служба, доступная через интернет. Web-сайт — как частный случай web-сервисов.

Проект Golden Eye задумывался именно как web проект. То есть это должен быть сервис, доступный через интернет. На нем можно будет посмотреть курсы валют, которые хранятся в базе данных, получить курсы валют через api запрос — в xml или json формате. А также можно будет обновить курсы валют из сторонних источников или изменить вручную.

Python Practice

Задачи каждого web сервиса

Каждое web приложение, независимо от своей бизнес-логики, должно выполнять определенные функции:

- Принимать HTTP запросы.
- Формировать ответы.
- Работать с куками, сессиями.
- Осуществлять роутинг url.
- Аутентификация пользователей.

Python Practice

Фреймворки спешат на помощь!

Для упрощения разработки, чтоб для каждого web приложения не приходилось реализовывать весь этот функционал на таком техническом уровне — используются web фреймворки.

Фреймворк (от англ. framework) — набор библиотек, которые предоставляют общие шаблоны для построения надежных, масштабируемых и поддерживаемых web-приложений. И разработчику остается только реализовать бизнес логику web приложения.

Несмотря на то, что использование web фреймворка не является обязательным условием для построения web приложения, очень редко разработчик не использует их для ускорения разработки.

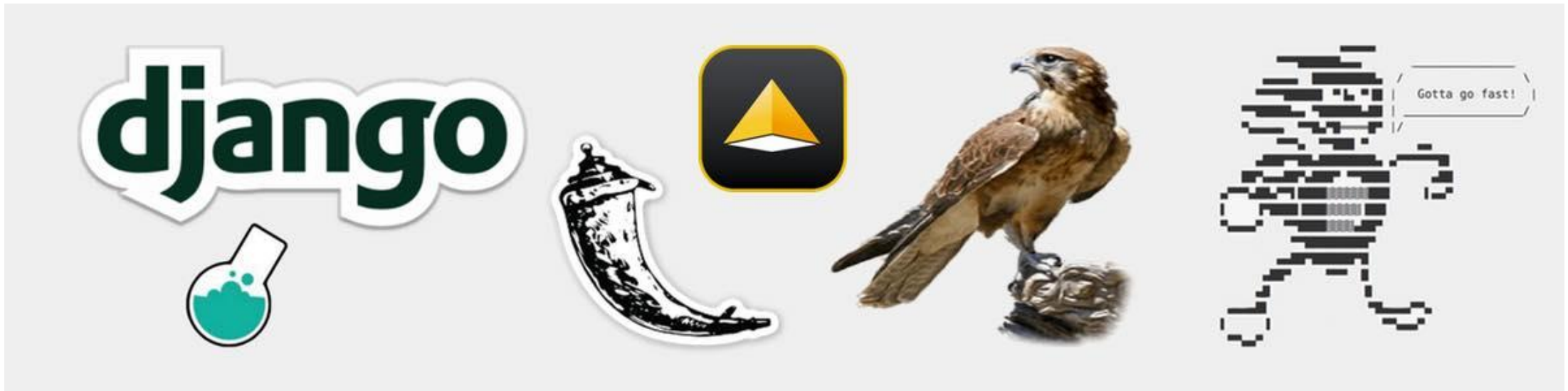
Python Practice

Фреймворки на Python

В данный момент существует большое количество web фреймворков, написанных на Python.

Наиболее популярные сейчас — это **Django** и **Flask**.

Оба они позволяют создавать проекты различной сложности — от простейших сайтов до крупных web проектов со сложной логикой, расширенным функционалом. Но каждый из них реализован в разной концепции, философии.



Python Practice

Django VS Flask

django

Философия "batteries-included"

Содержит весь необходимый функционал для построения приложения — от роутинга url до orm библиотеки доступа к БД, работы с сессиями.

Достаточно установить фреймворк и можно создавать полноценное приложение.

Но если мы захотим использовать другую orm библиотеку, например, то это может стать большой проблемой, если вообще возможно.



Flask

Философия extensibility, расширяемости

Включает только небольшое ядро с минимальным функционалом, но зато очень легко расширяемым.

Можно устанавливать нужные модули (расширения) и использовать их.

Python Practice

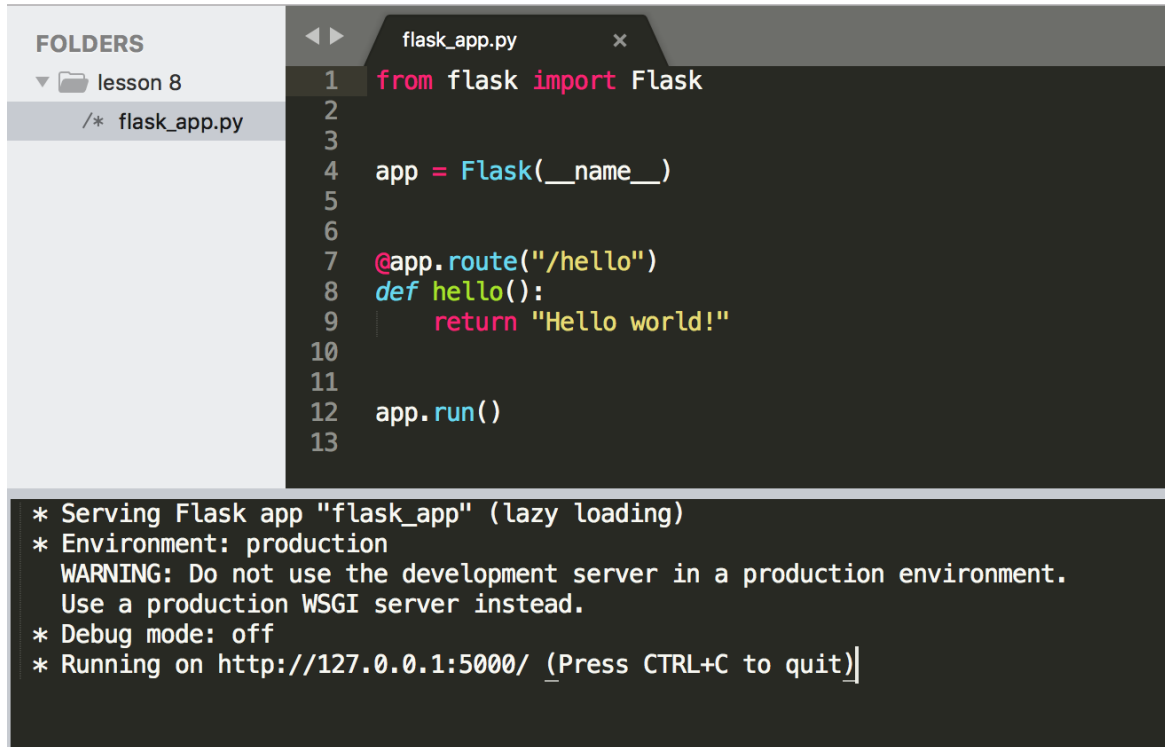
Создание web приложения с помощью Flask

Прежде чем использовать Flask фреймворк, необходимо установить его в окружение для проекта — с помощью команды `pip install flask==1.0.2` (с указанием версии).

Для создания простейшего web приложения на основе Flask фреймворка достаточно создать один(!) python модуль, в котором будет несколько строчек кода — импорт класса Flask, создание экземпляра этого класса — это и будет наше Flask приложение. Также необходимо описать функцию и в декораторе описать url, вызов которого приведет к вызову описанной функции. И вызвать метод `run` у экземпляра класса Flask, нашего приложения. При этом приложение будет запущено на локальном сервере, и указанный url будет доступен в браузере!

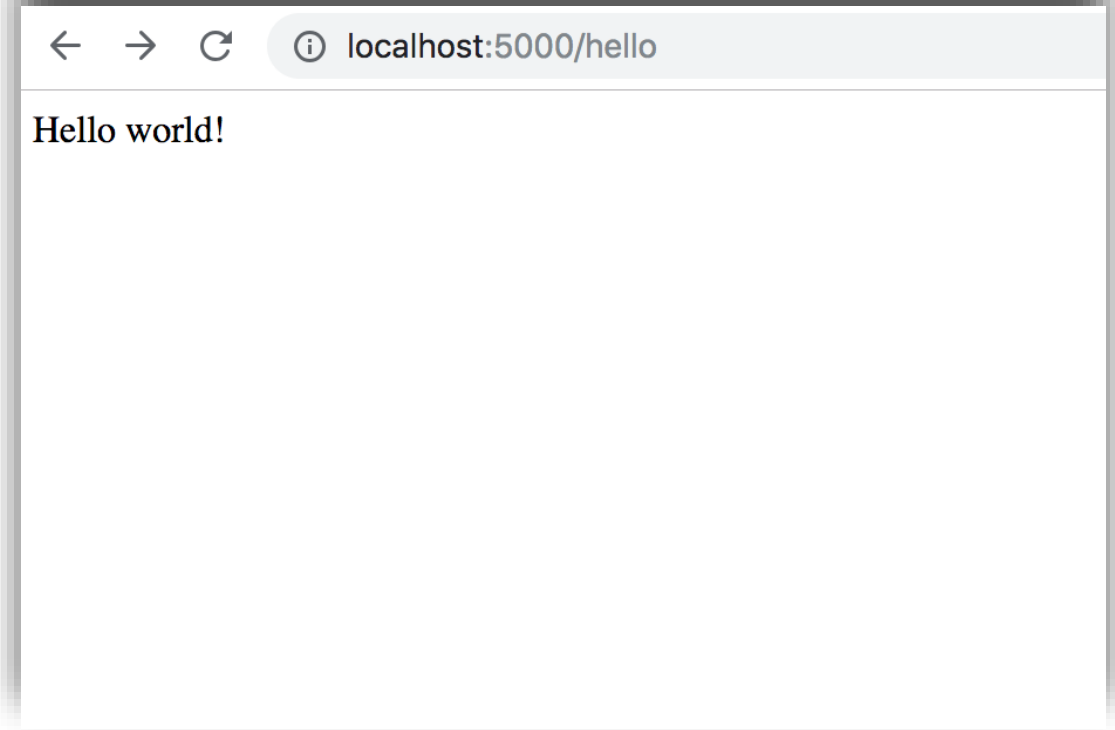
Python Practice

Web приложение на Flask



The screenshot shows a code editor with a file named `flask_app.py`. The code defines a Flask application with a single route `/hello` that returns "Hello world!". The application is running on `http://127.0.0.1:5000/`. The output console shows the following messages:

```
* Serving Flask app "flask_app" (lazy loading)
* Environment: production
  WARNING: Do not use the development server in a production environment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```



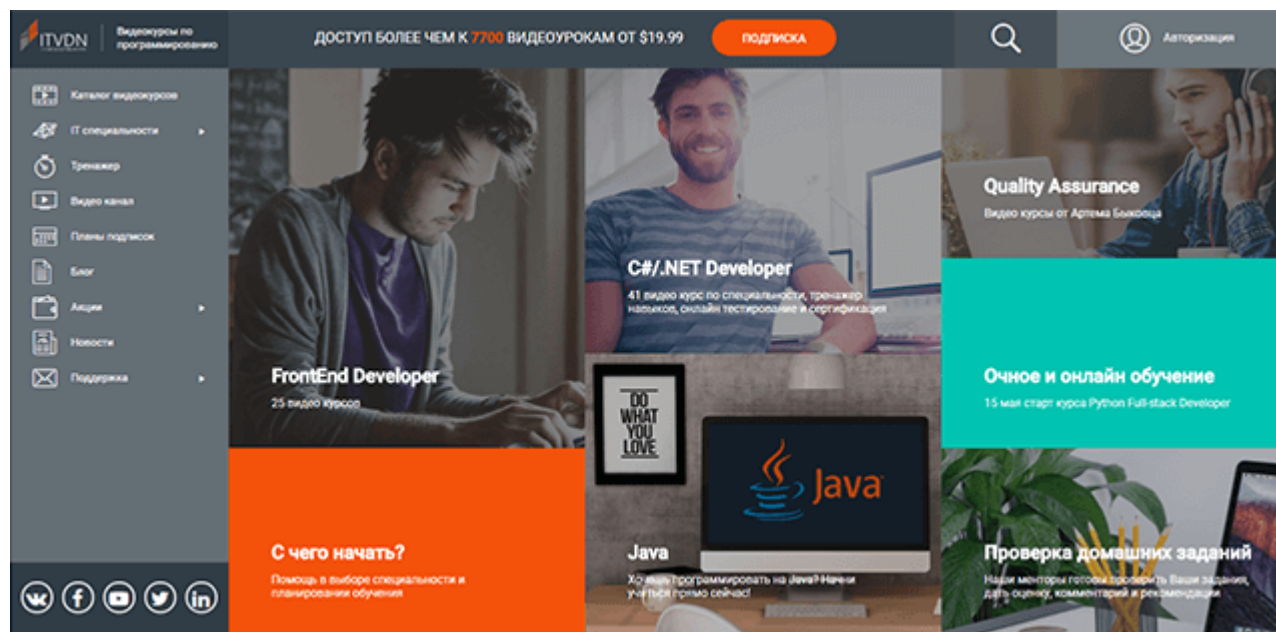
Python Practice

Этапы реализации web части проекта

- Реализация отображения курсов валют.
- Добавление api методов для получения данных о курсах по api.
- Добавление возможности обновления курсов из удаленных источников.
- Редактирование курсов вручную.
- Автоматическое обновление курсов валют.
- Размещение сервиса на Heroku.

Смотрите наши уроки в видео формате

ITVDN.com



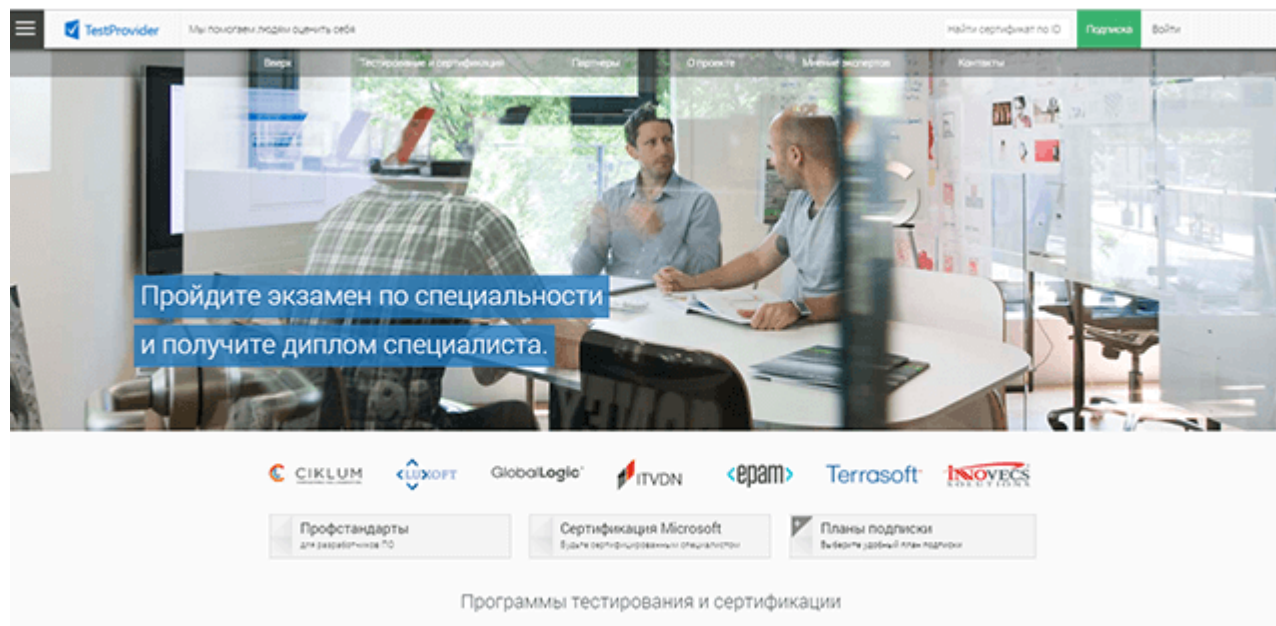
Посмотрите этот урок в видео формате на образовательном портале ITVDN.com для закрепления пройденного материала.

Курсы записаны сертифицированными тренерами, которые работают в учебном центре CyberBionic Systematics и другими высококвалифицированными разработчиками.



Проверка знаний

TestProvider.com



TestProvider – это online сервис проверки знаний по информационным технологиям. С его помощью Вы можете оценить Ваш уровень и выявить слабые места. Он будет полезен как в процессе изучения технологии, так и для общей оценки знаний IT специалиста.

После каждого урока проходите тестирование для проверки знаний на [TestProvider.com](https://testprovider.com)

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.



Python Practice

Q&A

Информационный видеосервис для разработчиков программного обеспечения

