ТЕХНОТРЕК

Лекция 10

Обзор фреймворков. Задачи на собеседовании.

Всё получится)

Что нужно знать

https://roadmap.sh/backend

Полезные библиотеки и инструменты

https://github.com/cookiecutter/cookiecutter

https://www.crummy.com/software/BeautifulSoup/bs4/doc/

https://github.com/mozilla/bleach

https://github.com/grey0ne/django-protector

https://github.com/Fantomas42/django-tagging

https://github.com/python-pillow/Pillow

https://gitlab.com/doctormo/python-crontab/

https://github.com/mnooner256/pyqrcode

https://github.com/python-telegram-bot/python-telegram-bot

https://github.com/jazzband/tablib

https://pypi.org/project/xlwt/

https://sentry.io/welcome/

Веб-фреймворки Python

Flask

- есть встроенный сервер и дебаггер
- шаблонизатор jinja2
- поддержка безопасных cookies
- возможность подключения любой ORM

Flask

Плюсы

- понятный, минималистичный, легковесный, модульный
- быстрое прототипирование
- доступно множество библиотек для подключения

Минусы

- относительно низкоуровневый

Где используется

- Rainist - онлайн-платформа для управления личными финансами
- Netflix - одна из крупнейших стриминговых платформ в мире
- Lyft - платформа для заказа такси и аренды транспорта

Flask. Пример приложения

```
from flask import Flask, escape, request

app = Flask(__name__)

@app.route('/')
def hello():
    name = request.args.get("name", "World")
    return f'Hello, {escape(name)}!'



# запуск
$ env FLASK_APP=hello.py flask run
```

Pyramid

- однофайловые приложения
- генерация URL
- масштабируемая конфигурация
- гибкая схема аутентификации и авторизации
- доступная пониманию техническая документация

Pyramid

Плюсы

- гибкость и удобство кастомизации
- ajax-запросы
- поддержка SQLAlchemy

Минусы

- требует времени на развёртывание и подготовку к разработке.
- надо хорошо разбираться в Pyramid

Pyramid. Пример приложения

```
from wsgiref.simple_server import make_server
from pyramid.config import Configurator
from pyramid.response import Response

def hello_world(request):
    return Response('Hello World!')

if __name__ == '__main__':
    with Configurator() as config:
        config.add_route('hello', '/')
        config.add_view(hello_world, route_name='hello')
        app = config.make_wsgi_app()
    server = make_server('0.0.0.0', 6543, app)
    server.serve_forever()
```

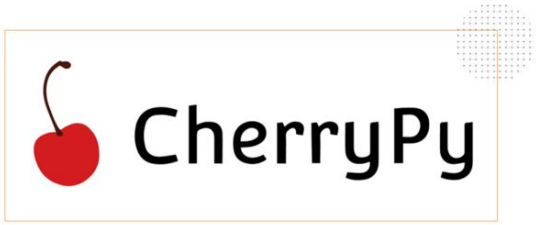
Bottle

- мини-фреймворк для API и прототипирования

Bottle. Пример приложения

```
from bottle import route, run, template

@route('/hello/<name>')
def index(name):
    return template('<b>Hello {{name}}</b>!', name=name)

run(host='localhost', port=8080)
```

Cherry Py



- простой запуск нескольких HTTP-серверов одновременно
- мощная конфигурационная система;
- гибкая система плагинов
- возможности «из коробки»: кэширование, декодирование, сессии, аутентификация, статический контент и т.п.
- возможность работы под Python 2.7+, Python 3.1+, PyPy, Jython и Android

Cherry Py. Пример приложения

```
import cherrypy

class HelloWorld(object):
    @cherrypy.expose
    def index(self):
        return "Hello World!"

cherrypy.quickstart(HelloWorld())
```

http://www.defuze.org/archives/228-running-cherrypy-on-android-with-sl4a.html

Асинхронное программирование

- процессорное переключение контекста;
- состояние гонки;
- взаимная/активная блокировка;
- исчерпание ресурсов.

Tornado

Асинхронный фреймворк, благодаря неблокирующему обмену данными способный одновременно поддерживать множество пользовательских соединений. Прекрасно подходит для задач, требующих подолгу поддерживать соединение с каждым пользователем.

Tornado

Плюсы

- Есть свой механизм аутентификации, при необходимости можно подключить сторонние.
- Поддержка переводов и локализации.
- Работа в реальном времени.

Минусы

- Меньшая популярность (чем у дженго).
- Сложность кода.

Где используется

- Uploadcare — облачный сервис для работы с файлами.

Tornado. Пример приложения

```
class MainHandler(tornado.web.RequestHandler):
    def get(self):
        self.write("You requested the main page")

class StoryHandler(tornado.web.RequestHandler):
    def get(self, story_id):
        self.write("You requested the story " + story_id)

application = tornado.web.Application([
     (r"/", MainHandler),
     (r"/story/([0-9]+)", StoryHandler),
])
```

Twisted

Twisted — это сетевой фреймворк, написанный на Python. Он поддерживает множество протоколов, имеет модули для создания web-серверов, чат-клиентов и серверов, почтовых серверов, ssh-серверов и т. д.

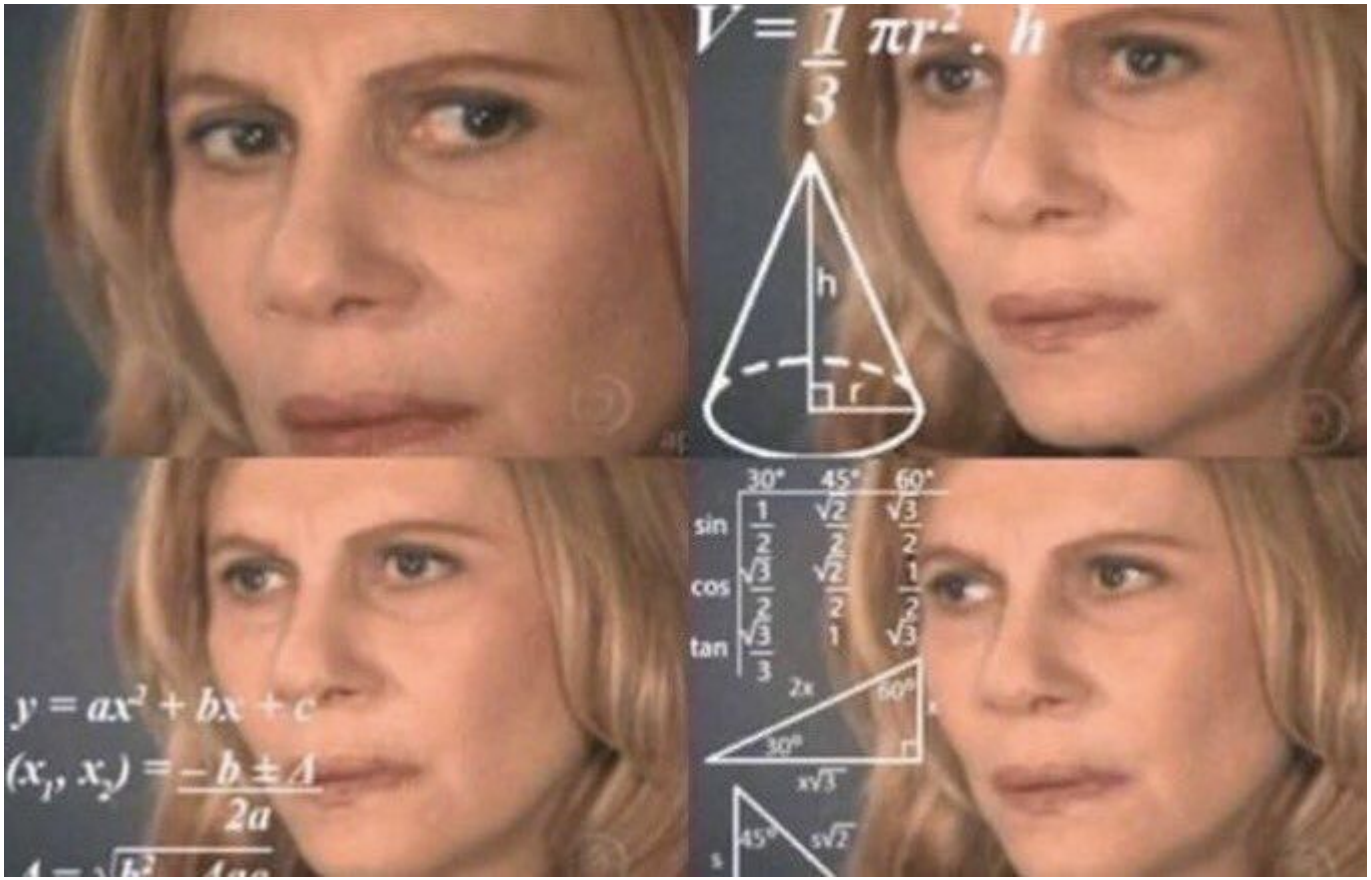
Twisted. Пример приложения

```
from twisted.internet import protocol, reactor

class Twist(protocol.Protocol):

    def connectionMade(self):
        print 'connection success!'

    def dataReceived(self, data):
        print data
        #transport.write - отправка сообщения
        self.transport.write('Hello from server!')

    def connectionLost(self, reason):
        print 'Connection lost!'

factory = protocol.Factory()
factory.protocol = Twist
print 'wait...'
reactor.listenTCP(777, factory)
reactor.run()
```

Углубленный питон. c-extensions

1. Вам нужна скорость и вы знаете, что C в X раз быстрее Python;
2. Вам нужна конкретная C-библиотека и вы не хотите писать “велосипед” на Python;
3. Вам нужен низкоуровневый интерфейс управления ресурсами для работы с памятью и файлами;
4. Просто потому что Вам так хочется.

Время задачек



Домашнее задание

- Настало время доделать все домашние работы за семестр

ТЕХНОТРЕК

Спасибо за внимание!