Python Practice

Реализация первых методов api

Автор курса

Кременtарь Ксения

Ведущий Python разработчик

Системный архитектор

в компании K-Solutions

После урока обязательно

Повторите этот урок в видео формате на ITVDN.com

Проверьте как Вы усвоили данный материал на TestProvider.com

Реализация первых методов api

На этом уроке

1. Узнаем, что такое api метод.
2. Создадим api метод для получения информации о курсах валют в различных форматах - json или xml.
3. Научимся создавать url с динамической частью во Flask приложениях.
4. Научимся обрабатывать GET запрос, извлекать его аргументы.
5. Напишем тесты для проверки работы первого api метода.

API методы web приложения

У нас уже есть web приложение, которое умеет обрабатывать запрос и возвращать html для отображения в браузере.

Но что если мы хотим сделать наше приложение доступным для других программ, клиентов в Интернете. И возвращать информацию о курсах валют, всем, кому она может быть нужна, но не в html формате.

Для этого нам достаточно добавить новую точку входа, или еще говорят, новый api метод, цель которого — возвращать информацию о курсах валют.

Мы сами можем придумывать протокол, то есть правила взаимодействия с этим api методом, так как мы являемся разработчиками этого api метода.

На самом деле, мы можем придумывать столько api методов, сколько нам будет угодно!

Логика работы api метода

Выберем название нашего метода, а именно url (точка входа), при вызове которого будет происходить формирование информации о курсах валют.
Так как нужно возвращать информацию в xml формате или в json формате, то пусть будет две точки входа.
Например, /api/xrates/json и /api/xrates/xml

Также в качестве get аргументов могут быть переданы параметры from_currency, to_currency для уточнения курса валют, который интересует клиента. Необходимо предусмотреть обработку этих аргументов.

Но логика работы этих api методов будет не очень сильно отличаться между собой, по крайней мере логика получения информации из БД точно будет общей, и чтоб не писать много повторяющегося кода, можно использовать только одну view-функцию, но при этом формат возвращаемой информации будет зависеть от того, что именно ожидает получить клиент — json или xml.

Динамические url во Flask

Во Flask есть возможно указывать динамические url, то есть url с динамической частью, которая передается в качестве именованного аргумента во view функцию.

Синтаксис следующий: можно добавить секцию <variable_name> в url. И тогда в сигнатуре соответствующей view функции будет описан аргумент variable_name.

Дополнительно можно указывать конвертер, для уточнения типа аргумента.
Возможные типы конвертеров:

- string(по умолчанию)
- int
- float
- path
- uuid

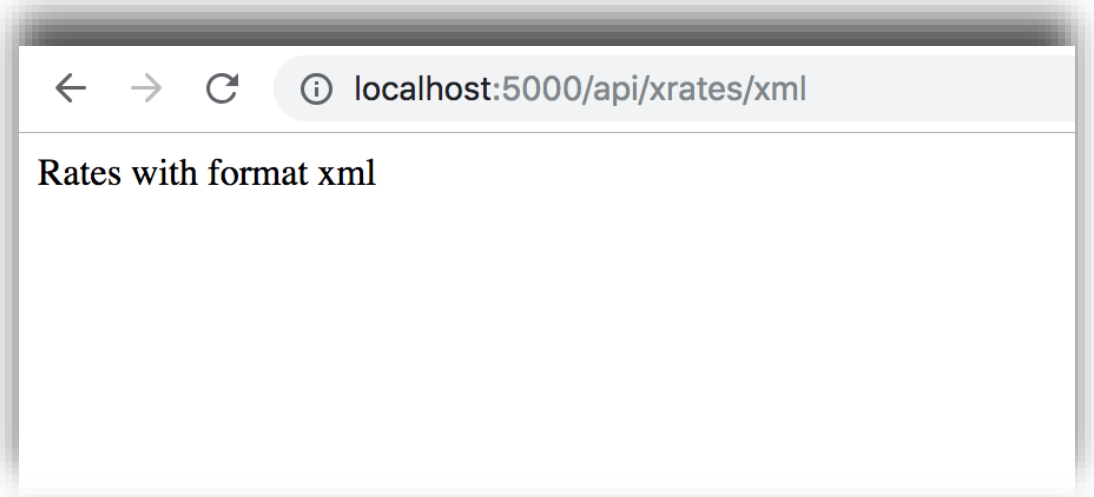
Числовые типы int, float — только положительные числа.

Примеры использования конвертеров

```
2  from flask import Flask
3
4
5  app = Flask(__name__)
6
7
8  @app.route("/hello")
9  def hello():
10     return "Hello world!"
11
12
13 @app.route("/hello/<name>")
14 def hello_name(name):
15     return f"Hello {name}!"
16
17
18 @app.route("/hello/<int:user_id>")
19 def hello_user_id(user_id):
20     return f"Hello {user_id}!"
21
22
23 @app.route("/hello/<float:height>")
24 def hello_height(height):
25     return f"Hello user with {height}!"
26
27
28 @app.route('/path/<path:subpath>')
29 def show_subpath(subpath):
30     # show the subpath after /path/
31     return 'Subpath %s' % subpath
32
33
34 app.run()
```

Функция api_rates

```
@app.route('/api/xrates/<fmt>')
def api_rates(fmt):
    return f"Rates with format {fmt}"
```



Обработка GET запросов во Flask

Для получения аргументов GET запроса во Flask приложении нужно воспользоваться параметром args объекта request Flask запроса.
Этот параметр возвращает объект типа werkzeug.datastructures.ImmutableMultiDict, основное отличие от обычных dict словарей — возможность добавлять несколько значений по одному ключу. Также ImmutableMultiDict, что следует из его названия, недоступен для редактирования, только для чтения значений и ключей.

С помощью параметра args можно проверить, переданы ли GET аргументы from_currency, to_currency и если переданы, получить их значения и использовать при получении курсов из БД.

Особенности реализации api метода

Модуль controllers содержит функцию get_all_rates, задача которой, помимо получения информации о курсах из БД и формирования html страницы — также предусмотреть обработку ошибок, возникающих в приложении.

Такой подход с обработкой ошибок очень удобен, и подойдет для обработки любого запроса. Поэтому есть смысл немного пересмотреть архитектуру кода и добавить новый класс контроллера, BaseController, в котором будет реализована общая логика — обработка ошибок, формирование логгера и т.п.

Для обработки каждого типа запросов будет написан свой контроллер-наследник класса BaseController, в котором будет реализована только специфичная для данного api метода логика.

Примерный вид controllers.py

```python
from flask import render_template, make_response

from models import XRate


class BaseController:
    def __init__(self):
        pass

    def call(self, *args, **kwds):
        try:
            return self._call(*args, **kwds)
        except Exception as ex:
            return make_response(str(ex), 500)

    def _call(self, *args, **kwds):
        raise NotImplementedError("_call")


class ViewAllRates(BaseController):
    def _call(self):
        xrates = XRate.select()
        return render_template("xrates.html", xrates=xrates)

```

Тестирование api методов

Для тестирования нового метода api воспользуемся уже написанными тестами, добавим только тесты, где будет происходить вызов api метода /api/xrates/json и /api/xrates/xml с помощью библиотеки requests.

То есть в отличие от предыдущих тестов, мы не будем вызывать какую-то часть кода (как было при тестировании обновления курсов с помощью различных удаленных api). Мы будем вызывать api метод приложения Golden Eye как сторонние клиенты, и проверять, какой ответ нам приходит.

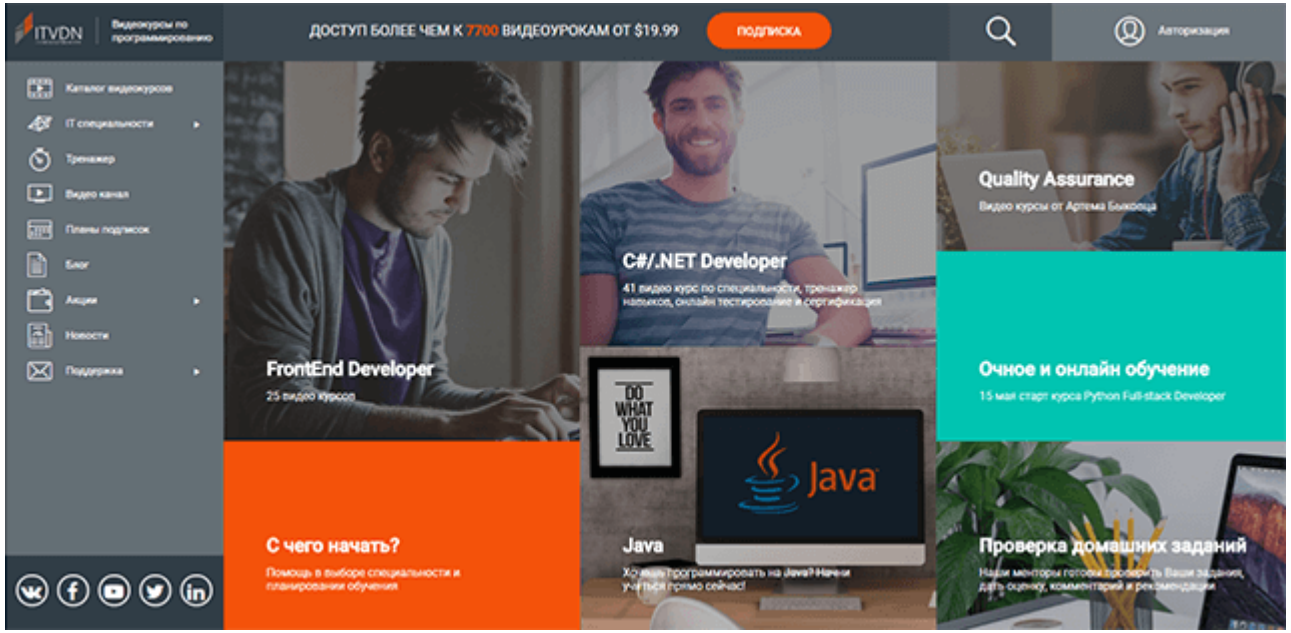
Что дальше?

Добавление возможности обновления курсов из удаленных источников — на странице с отображением курсов валют в виде html страницы.

То есть добавим html кнопку или ссылку, при нажатии на которую, будет происходить обновление курсов валют из удаленных источников, которые мы реализовывали в первой части курса, в пакете api.

Смотрите наши уроки в видео формате

ITVDN.com



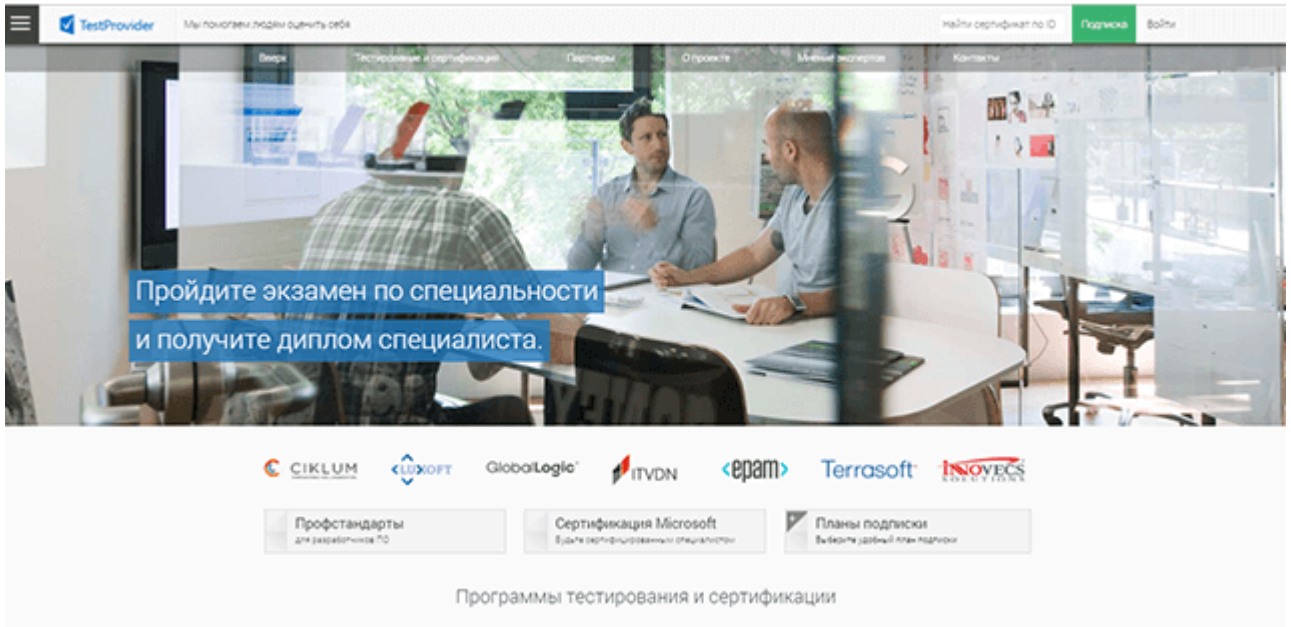
Посмотрите этот урок в видео формате на образовательном портале ITVDN.com для закрепления пройденного материала.

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

ITVDN
IT VIDEO DEVELOPERS NETWORK

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

Успешное прохождение финального тестирования позволит Вам получить соответствующий Сертификат.

TestProvider

Q&A

Информационный видеосервис для разработчиков программного обеспечения

ITVDN

IT VIDEO DEVELOPERS NETWORK

TestProvider PROMETRIC TEST CENTER CyberBionic systematics Microsoft Partner Silver Learning Windows Azure Cloud Partner EBA European Business Association