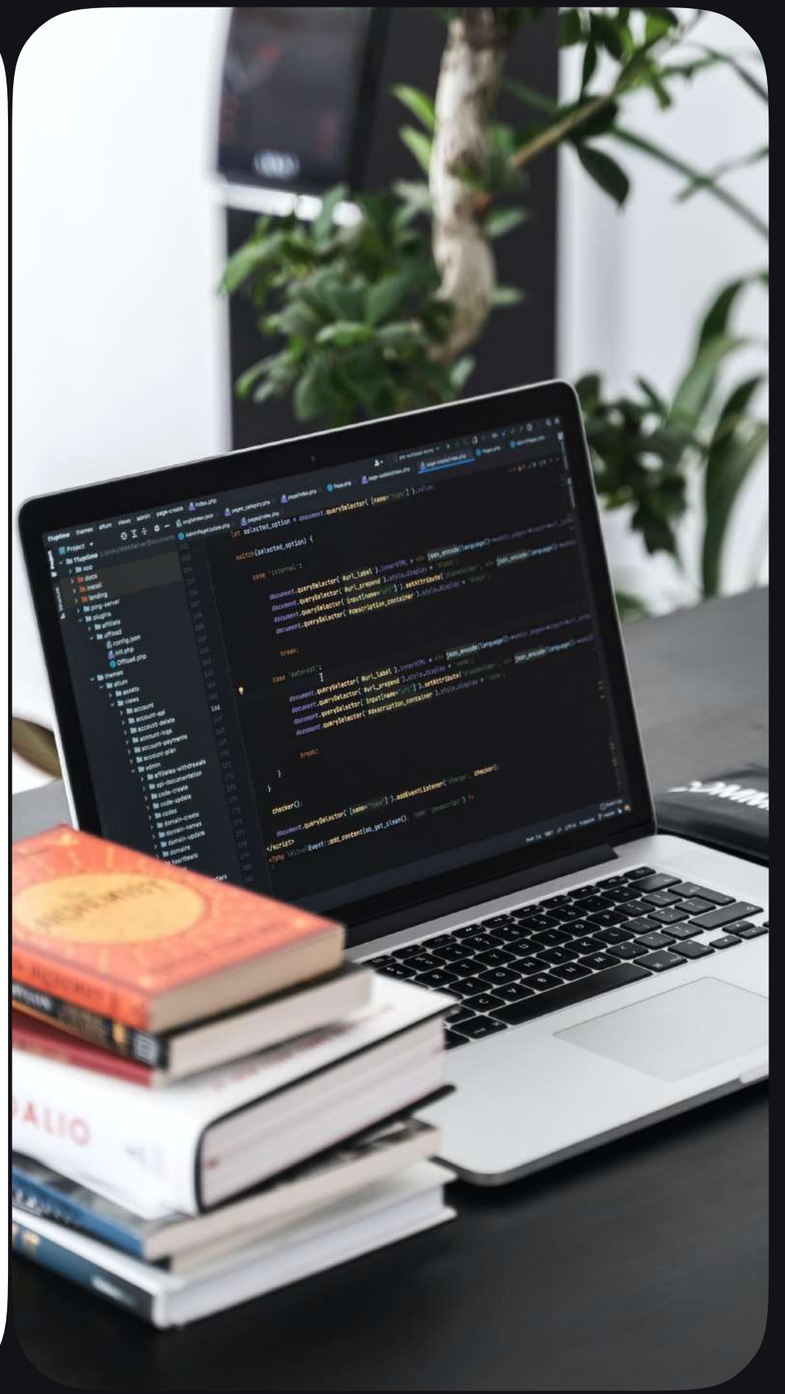
Модуль 3 Занятие 9

API



API

REST API

Выполнение запросов

Работа с API

Тестирование

API (Application Programming Interface, программный интерфейс приложения) — набор способов и правил, которые определяют, как программы обмениваются данными между собой.

Клиент хочет купить товар →

← Все в порядке, пусть покупает

Форматы передачи данных

Благодаря API программы могут общаться между собой, используя протокол HTTP и передавая данные в удобном формате, например:

- ✓ JSON (JavaScript Object Notation) — стандартный текстовый формат обмена данными.
- ✓ XML (eXtensible Markup Language) — расширяемый язык разметки. Похож на HTML, но названия тегов в XML могут быть произвольными.

JSON

Информация о текущей погоде в формате JSON:

JSON (JavaScript Object Notation) — стандартный текстовый формат обмена данными.

JSON по структуре похож на словарь в Python, но более стандартизирован.


```
{
    "coord": {
        "lon": 37.6191,
        "lat": 55.7592
    },
    "weather": [
        {
            "id": 600,
            "main": "Snow",
            "description": "небольшой снег",
            "icon": "13d"
        }
    ],
    "base": "stations",
    "main": {
        "temp": -2.85,
        "feels_like": -7.92,
        "temp_min": -4.23,
        "temp_max": -2.68,
        "pressure": 1012,
        "humidity": 96,
        "sea_level": 1012,
        "grnd_level": 994
    },
    "visibility": 480,
    "wind": {
        "speed": 4.18,
        "deg": 272,
        "gust": 10.8
    },
    "snow": {
        "1h": 0.13
    },
    "clouds": {
        "all": 100
    },
    "name": "Москва"
}
```

XML

XML (eXtensible Markup Language) — расширяемый язык разметки. Похож на HTML, но названия тегов в XML могут быть произвольными: теги пишутся в треугольных скобках, есть открывающий тег и закрывающий его тег.

```
<?xml version="1.0" encoding="UTF-8"?>
<current>
    <city id="524901" name="Москва">
        <coord lon="37.6173" lat="55.7558"></coord>
        <country>RU</country>
        <timezone>10800</timezone>
        <sun rise="2023-02-13T04:59:58" set="2023-02-13T14:27:44"></sun>
    </city>
    <temperature value="-3.05" min="-4.25" max="-2.7" unit="celsius"></temperature>
    <feels_like value="-8.14" unit="celsius"></feels_like>
    <humidity value="96" unit="%"></humidity>
    <pressure value="1012" unit="hPa"></pressure>
    <wind>
        <speed value="4.14" unit="m/s" name="Gentle Breeze"></speed>
        <gusts value="10.96"></gusts>
        <direction value="261" code="W" name="West"></direction>
    </wind>
    <clouds value="100" name="пасмурно"></clouds>
    <visibility value="672"></visibility>
    <precipitation mode="no"></precipitation>
    <weather number="600" value="небольшой снег" icon="13d"></weather>
    <lastupdate value="2023-02-13T09:14:05"></lastupdate>
</current>
```

REST

REST (REpresentational State Transfer, передача состояния представления) — это набор принципов взаимодействия компьютерных систем, с помощью протокола HTTP.

Основное понятие в REST — ресурс. Ресурсом может быть что угодно, к чему разработчик REST API считает важным предоставить доступ клиенту. Ресурс — это информация, которую приложение предоставляет клиентам.

Client

REST API

Server

REST API

REST — это набор принципов, которых следует придерживаться при создании API. Если API сделан по этим принципам, его называют RESTful API (или просто REST API).

Принципы REST:

1. Клиент-сервер
2. Отсутствие состояния
3. Единообразие интерфейса
4. Многоуровневость
5. Кешируемость
6. Код по запросу

Ресурсы

Основное понятие в REST — ресурс. Ресурс — это информация, которую приложение предоставляет клиентам. Каждый ресурс на сервере имеет свой уникальный URL-адрес — эндпоинт.

Например:

https://swapi.dev/api/people/	ресурс со списком людей
https://swapi.dev/api/people/1/	пользователь с id = 1 — тоже ресурс
https://swapi.dev/api/starships/	ресурс со списком кораблей

HTTP-методы

API предоставляет доступ к ресурсам по URL. При этом HTTP-метод запроса определяет, что следует сделать:

GET	получить информацию о ресурсе
POST	добавить ресурс
PUT	заменить существующий ресурс
PATCH	частично изменить существующий ресурс
DELETE	удалить ресурс

Выполнение запросов

Для выполнения сетевых запросов в Python, будем использовать модуль requests.

Для того, чтобы начать использовать модуль, выполните его установку, выполнив команду в терминале:

```
pip install requests
```

Пример

Выполнить get запрос можно с помощью одноименного метода get() модуля requests.

В результате мы получим объект класса Response — этот объект содержит ответ сервера. Текст ответа можно получить из свойства response.text.

```
import requests


response = requests.get('https://uchi.ru/')
response.encoding = 'utf-8'
print(type(response), response)
print(response.text)
```

Пример

```
import requests


response = requests.get('https://swapi.dev/api/planets/10/')
print(response)
text = response.text
print(type(text), text)
json = response.json()
print(type(json), json)
```

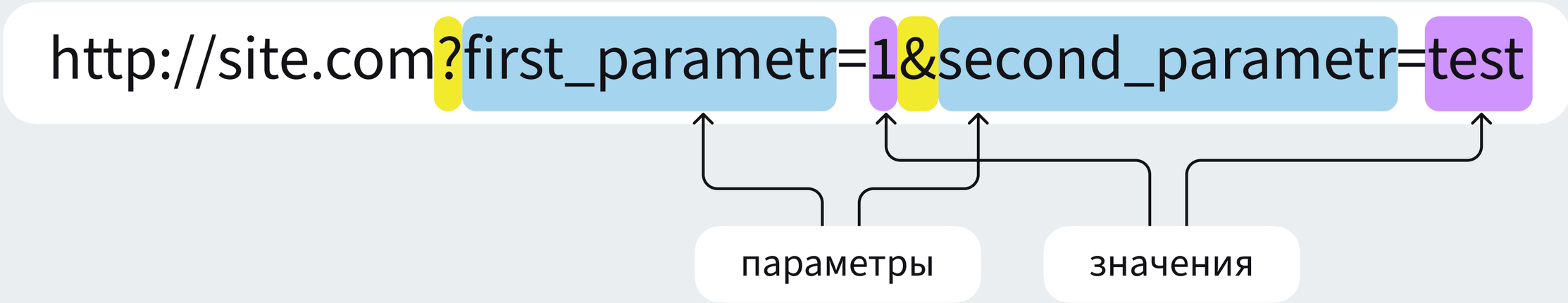
Преобразовать JSON-строку в тип данных Python, можно методом json() класса Response.

Вывод:

```
<Response [200]>
<class 'str'> {"name":"Kamino","rotation_period":"27", ...}
<class 'dict'> {'name': 'Kamino', 'rotation_period': '27', ...}
```

Запросы с параметрами

Запрос может содержать параметры — это дополнительная информация, которую можно добавить в URL-адрес. Параметры указываются в URL после знака **?**. Несколько параметров отделяются друг от друга знаком амперсанд **&**. Сам параметр состоит из двух обязательных элементов: из названия и его значения, разделенных знаком **=**.



Пример

```
import requests


params = {
    'lang': 'ru',
    'format': 'j1'
}


response = requests.get(
    'https://wttr.in/Moscow',
    params=params
)
response.encoding = 'utf-8'
print(response.url)
print(response.text
```

Для того, чтобы не составлять длинный URL самостоятельно, перечисляя все параметры, можно воспользоваться аргументом params метода get() и передать этому аргументу словарь с необходимыми параметрами. Метод get() выполнит все преобразования и сформирует URL-адрес.

Вывод:

```
https://wttr.in/Moscow?lang=ru&format=j1
```

Итоги

- API — набор способов и правил, которые определяют как программы обмениваются данными между собой
- REST API — это набор принципов, которых следует придерживаться при создании API
- Основное понятие в REST — ресурс. Ресурс — это информация, которую приложение предоставляет клиентам. Каждый ресурс на сервере имеет свой уникальный URL-адрес — эндпоинт
- Для выполнения сетевых запросов в Python можно использовать модуль requests

Как выполнять сетевые запросы в Python узнал ты и работать с API научился. Впереди перерыв ждет тебя, после которого продолжится обучения путь.

OAuth

Авторизация — предоставление пользователю прав на выполнение каких-либо действий.

Аутентификация — процедура проверки подлинности пользователя.

OAuth (Open Authorization) — схема (протокол) авторизации, обеспечивающая предоставление третьей стороне ограниченного доступа к защищенным ресурсам без передачи ей (третьей стороне) логина и пароля. Для этого используется OAuth-токен.

Токен обычно выглядит как уникальная последовательность символов, например так:

b1c2ac72fade9437189f65cb1c2ac72fade9437189f65c

Итоги

- **Авторизация** — предоставление пользователю прав на выполнение каких-либо действий
- **Аутентификация** — процедура проверки подлинности пользователя
- **OAuth** — протокол авторизации, обеспечивающая предоставление третьей стороне ограниченного доступа к защищенным ресурсам без передачи ей (третьей стороне) логина и пароля. Это становится возможным благодаря OAuth-токену