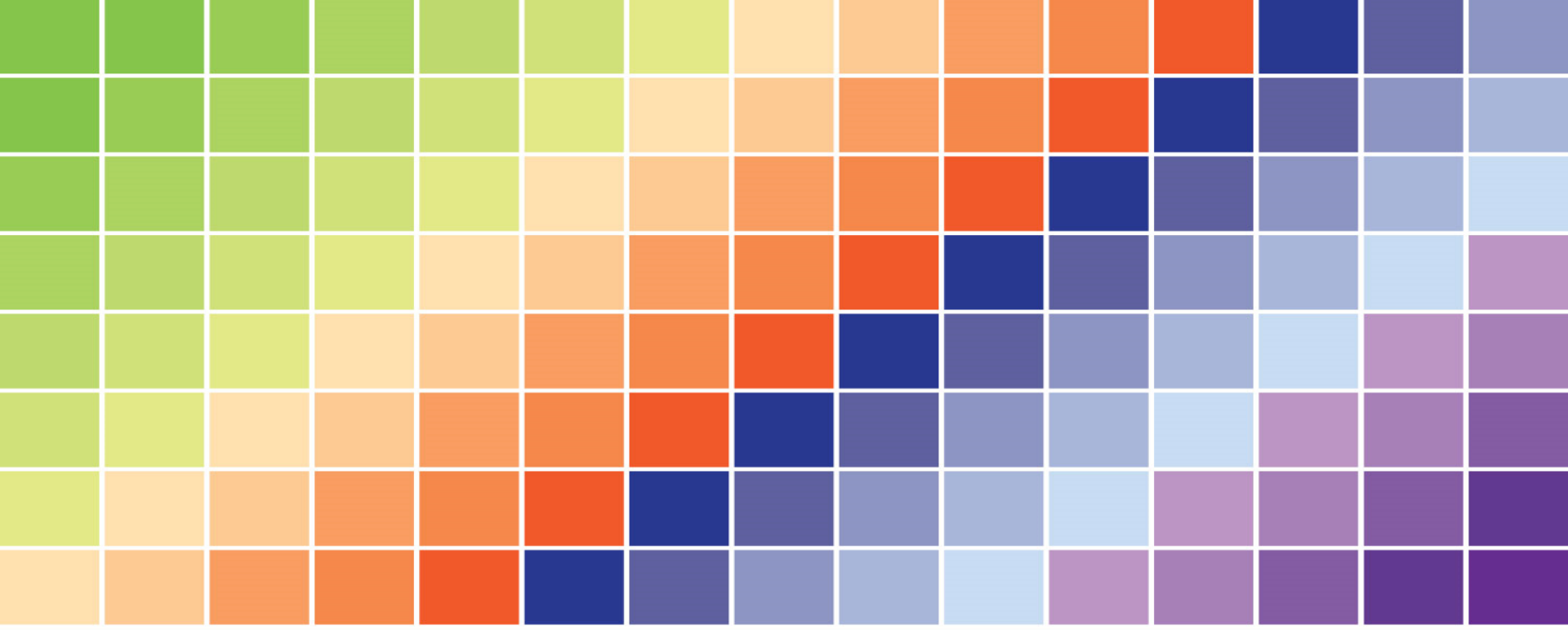


School of Computer Science

REQUESTS И WEB API. ПРАКТИЧЕСКИЕ ПРИМЕРЫ

ПРОГРАММИРОВАНИЕ НА PYTHON

Лекции для IT-школы

Что такое HTTP запросы
БИБЛИОТКА REQUEST

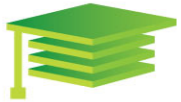
- HTTP(***HyperText Transfer Protocol***) **-** это протокол передачи данных, используемый для обмена информацией между клиентом и сервером

- Запросы(**Request**): способ общения компьютеров в сети интернет. Они позволяют получать данные с удаленных серверов или отправлять данные на них по протоколу HTTP

Что такое HTTP запросы
БИБЛИОТКА REQUEST

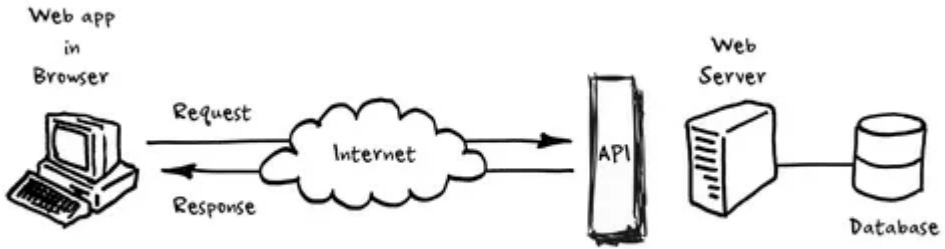
- HTTP(***HyperText Transfer Protocol***) **-** это протокол передачи данных, используемый для обмена информацией между клиентом и сервером

- Запросы(**Request**): способ общения компьютеров в сети интернет. Они позволяют получать данные с удаленных серверов или отправлять данные на них по протоколу HTTP



Что такое HTTP запросы
БИБЛИОТКА REQUEST

- В ответ на запрос сервер получает ответ(**Response**) это то, что программа получает после отправки запроса к серверу. Это может быть любая информация, которую сервер отсылает обратно клиенту в ответ на запрос.



Основные принципы HTTP запросов
БИБЛИОТКА REQUEST

- **Клиент-Серверная архитектура:** HTTP работает по модели клиент-сервер, где клиент (например, веб-браузер) отправляет запросы, а сервер отвечает на них.
- **Без состояния (Stateless):** Протокол HTTP не сохраняет состояние между запросами, каждый запрос обрабатывается независимо от предыдущих.
- **Методы запросов:** HTTP определяет различные методы запросов, такие как GET, POST, PUT, DELETE, которые определяют тип действия, выполняемого над ресурсом на сервере.
- **Единый протокол:** Взаимодействие между клиентом и сервером должны осуществляться по единому протоколу

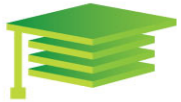
МЕТОДЫ HTTP
БИБЛИОТКА REQUEST

- **GET:** Используется для получения данных с сервера. Например, загрузка веб-страницы или изображения.
- **POST:** Используется для отправки данных на сервер. Например, отправка данных формы или создание нового ресурса на сервере.
- **PUT:** Используется для обновления данных на сервере. Например, обновление информации о пользователе.
- **DELETE:** Используется для удаления данных на сервере. Например, удаление файла или записи из базы данных.

СТАТУСНЫЕ КОДЫ
БИБЛИОТКА REQUEST

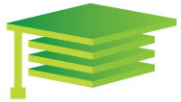
Статусные коды являются ключевой частью коммуникации между клиентом и сервером, помогая понять результат выполнения запроса и принять соответствующие действия.

Код ответа	Описание	Пример
1xx - Информационные коды	Коды этой категории информируют о процессе обработки запроса, но не являются окончательными ответами	Пример: 100 Continue (Продолжайте).
2xx - Успешные коды	Коды этой категории указывают на успешное выполнение запроса.	Пример: 200 OK (Успешно).



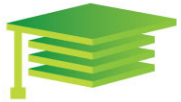
СТАТУСНЫЕ КОДЫ
БИБЛИОТКА REQUEST

Код ответа	Описание	Пример
3xx - Перенаправление	Коды этой категории информируют о процессе обработки запроса, но не являются окончательными ответами	Пример: 301 Moved Permanently (Перемещено навсегда).
4xx - Ошибки клиента:	Коды этой категории указывают на ошибки, вызванные некорректным запросом со стороны клиента.	Пример: 404 Not Found (Не найдено).
5xx - Ошибки сервера:	Коды этой категории указывают на ошибки, связанные с невозможностью сервера выполнить запрос.	Пример: 500 Internal Server Error (Внутренняя ошибка сервера)



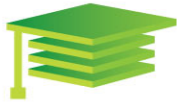
ИЗ ЧЕГО СОСТОИТ ТЕЛО ЗАПРОСА
БИБЛИОТКА REQUEST

Код ответа	Описание	Пример
Метод (Method):	Определяет тип операции, которую клиент хочет выполнить над ресурсом на сервере.	GET, POST, PUT, DELETE
URI (Uniform Resource Identifier):	Указывает на адрес ресурса, к которому обращается запрос.	http:localhost:8065/api/users или https://example.com/data.



ИЗ ЧЕГО СОСТОИТ ТЕЛО ЗАПРОСА
БИБЛИОТКА REQUEST

Код ответа	Описание	Пример
HTTP Версия (HTTP Version):	Указывает на версию протокола HTTP, используемую в запросе.	Например: HTTP/1.1 или HTTP/2.
Заголовки (Headers):	Передают дополнительную информацию о запросе, такую как тип содержимого, дата и время, параметры безопасности и другие метаданные	Примеры заголовков: Content-Type, User-Agent, Accept
Тело запроса (Request Body):	Присутствует только в запросах с методами, которые отправляют данные на сервер, например, POST или PUT.	Содержит данные, которые клиент отправляет на сервер, например, форма с данными пользователя или JSON-объект.



ИЗ ЧЕГО СОСТОИТ ТЕЛО ОТВЕТА
БИБЛИОТКА REQUEST

Код ответа	Описание	Пример
Статусная строка (Status Line):	Строка, содержащая статусный код и соответствующее описание статуса выполнения запроса.	Пример: HTTP/1.1 200 OK
Заголовки (Headers):	Метаданные, содержащие дополнительную информацию о ответе, такую как тип содержимого, дата и время, параметры безопасности и другие метаданные	Content-Type: application/json
Тело ответа (Response Body):	Фактические данные или содержимое ответа, передаваемые от сервера клиенту.	Например, HTML-страница, JSON-данные, изображение, файл и т. д.

Основные принципы HTTP запросов
БИБЛИОТКА REQUEST

- API **(Application Programming Interface) –** это набор правил и протоколов, который позволяет разным программам взаимодействовать друг с другом.

- **Преимущества использования API**:
  1. Интеграция с внешними сервисами и приложениями
  2. Экономия времени и трудозатрат.
  3. Масштабируемость
  4. Возможность повторного использования

Основные принципы HTTP запросов
БИБЛИОТКА REQUEST

- API **(Application Programming Interface) –** это набор правил и протоколов, который позволяет разным программам взаимодействовать друг с другом.

- **Преимущества использования API**:
  1. Интеграция с внешними сервисами и приложениями
  2. Экономия времени и трудозатрат.
  3. Масштабируемость
  4. Возможность повторного использования

Основные принципы HTTP запросов
БИБЛИОТКА REQUEST

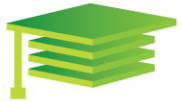
- **Эндпоинты (Endpoints):** URL-адреса, по которым можно отправлять запросы к API для выполнения определенных действий или получения данных.
- **Методы HTTP:** Определяют тип действия, выполняемого над ресурсом при запросе к API (GET, POST, PUT, DELETE и т. д.).
- **Форматы данных:** Обычно API возвращает данные в определенных форматах, таких как JSON, XML, текст и т. д.
- **Документация:** Обычно API хорошо документируются, если этого не делать, то пользователь будет посылать неправильные запросы

загрузка WEB-страниц
Requests.get(), выброс исключений

- В Python IDLE Shell введите код:

```
>>> import requests
>>> res = requests.get("http://compassplus.ru/Page_which_not_exists")
>>> res.raise_for_status()
Traceback (most recent call last):
  File "<pyshell#24>", line 1, in <module>
    res.raise_for_status()
  File "C:\Users\slukashenko\AppData\Local\Programs\Python\Python35-32\lib\
site-packages\requests\models.py", line 935, in raise_for_status
    raise HTTPError(http_error_msg, response=self)
requests.exceptions.HTTPError: 404 Client Error: Component not found for ur
l: http://compassplus.ru/Page_which_not_exists
```

- Метод raise_for_status() объекта res генерирует исключение HTTPError, если считывание страницы не было удачным



Исключения при загрузке из WEB
Requests.Get(), проверка исключений

- В Python IDLE Shell введите код:

```
>>> import requests
>>> res = requests.get("http://compassplus.ru/Page_which_not_exist")
>>>
try:
    res.raise_for_status()
except Exception as exc:
    print("Возникла проблема: {}".format(exc))

Возникла проблема: 404 Client Error: Component not found for url: http:
//compassplus.ru/Page_which_not_exist
```

- Целесообразно обрабатывать возможные исключения после вызова метода raise_for_status(), который следует сразу же за чтением web-страницы с помощью вызова requets.get() – см. пример в many_exc_demo.py

Интересные числа

Практическое задание

- С клавиатуры в список вводятся целые числа
- Для каждого числа:
  - Узнайте, существует ли интересный математический факт об этом числе
  - Выведите Interesting, если для числа существует интересный факт, иначе – Boring
  - Информация о числах выводится в том же порядке как они вводились
- Используем WEB API сайта www.numbersapi.com
- Формат запроса:

  http://numbersapi.com/<num>/math?json=true

  где <num> – проверяемое число

Интересные числа
Примеры

Пример введённых чисел:
31
999
1024
502

Пример вывода программы:
Interesting
Boring
Interesting
Boring

Примеры запросов:

По интересному числу:
http://numbersapi.com/7/math?json=true

По скучному числу:
http://numbersapi.com/999/math?json=true

использование Web API
пример JSON web-сервиса

- Типовые форматы сообщений: JSON, XML
- Пример запроса погоды по WEB API
  - https://openweathermap.org/api
  - См. скрипт в блокноте openweather.py

СПАСИБО ЗА ВНИМАНИЕ !

ВОПРОСЫ ?

School of Computer Science