

Основы клиент-серверного взаимодействия. Парсинг API

Урок 1



Петр Рубин

GeekBrains

Data Scientist, врач, к.м.н, преподаватель высшей школы

- ☀️ ex-декан факультета «DataScience в медицине» GeekBrains
- ☀️ занимался спектральным анализом сигналов лазерной доплеровской флоуметрии;

Stanford | ONLINE

Oct 30, 2020

PETR MIKHAILOVICH RUBIN

Учащийся успешно прошел(а) курс

Machine Learning

онлайн курс без права на зачетные единицы от университета Stanford University, предлагаемый на Coursera



Sep 13, 2020

PETR MIKHAILOVICH RUBIN

Учащийся успешно прошел онлайн-специализацию без права на зачетные единицы

Deep Learning

Andrew Ng,
Founder,
DeepLearning.AI

Kian Katanforoosh
Co-founder, Workera

Younes Bensouda



Dec 19, 2020

PETR MIKHAILOVICH RUBIN

has successfully completed the online, non-credit Professional Certificate

IBM Data Science

Ravi Ahuja
AI & Data Science
Program Director
IBM Skills Network



Garbage in, garbage out — «мусор на входе — мусор на выходе»



Garbage in



Идеальная модель



Garbage out



Плохие данные

- 💡 маленькая выборка (недостаточное количество данных)
- 💡 нерепрезентативная выборка
- 💡 несбалансированная выборка
- 💡 отсутствующие/пропущенные данные
- 💡 устаревшая информация
- 💡 данные введены в неправильное поле
- 💡 дублирование записей
- 💡 ошибки, опечатки и орфографические вариации



Специалисты, занимающиеся сбором и разметкой данных



Data Collector



Data Labeler



Data Engineer



Data Scientist



План курса

1

Основы клиент-серверного взаимодействия. Парсинг API.

2

Парсинг HTML. BeautifulSoup.

3

СУБД MongoDB и ClickHouse в Python

4

Парсинг HTML. XPath.

5

Scrapy.

6

Scrapy. Парсинг фото и файлов.

7

Selenium в Python.

8










Работа с данными.

9

Инструменты разметки наборов данных.



Что будет на уроке сегодня

-  Клиент-серверное взаимодействие
-  Введение в Web APIs
-  Протоколы прикладного уровня (Application) в OSI-модели
-  Конечные точки и запросы API.
-  Representational State Transfer (REST)
-  Клиент-серверный поток HTTP
-  Создание HTTP-запросов в Postman
-  Создание HTTP-запросов в Python
-  Создание датафрейма в Jupyter-ноутбуке.



Парсинг данных

— это процесс получения данных в одном формате и преобразования их в другой формат.





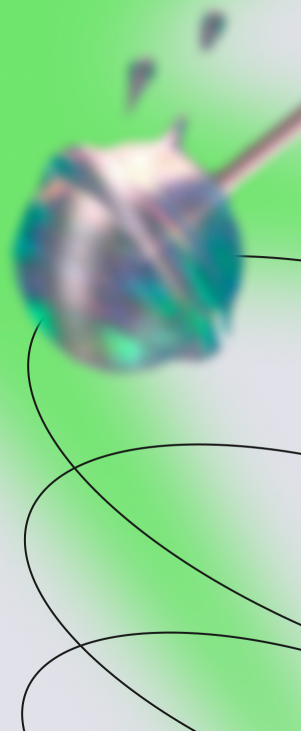
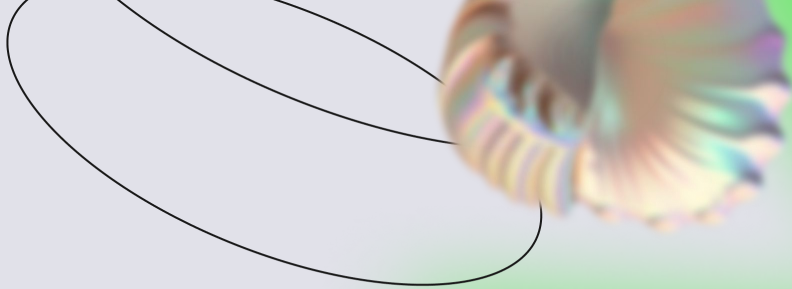
Веб-скрейпинг (или скрепинг, или скрапинг)

— это технология получения данных путем извлечения их со страниц веб-ресурсов.





Взаимодействие клиента и сервера





Цикл "запрос-ответ" (request-response cycle)



Клиент посылает запрос на сервер.



Сервер получает запрос и обрабатывает его.



Сервер отправляет ответ клиенту.



Клиент получает ответ и обрабатывает информацию.



Типы взаимодействия клиент-сервер



Простой запрос-ответ



Stateful взаимодействие



Взаимодействие в реальном времени



Основы Web APIs





Вопрос

Зачем нужна коммуникация
между программами?





API (Application Programming Interface)

— это описание способа коммуникации между двумя единицами кода, иными словами это способ связи между двумя компьютерными программами.





Типы API



Открытые API



Внутренние API



Партнерские API



Мэша́п (mashup)

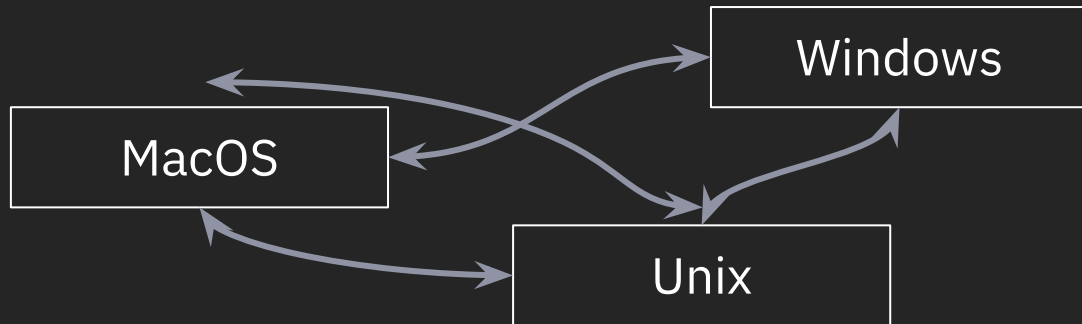
— это веб-приложение, объединяющее данные из нескольких источников в один интегрированный инструмент





Вопрос

Как мы можем быть уверены, что нам удастся объединить в одной программе сервисы разных компаний, которые используют разные операционные системы, спецификации и структуры данных?





Протокол

— это набор правил и действий (очерёдности действий), позволяющий осуществлять соединение и обмен данными между двумя и более включенными в сеть устройствами.



Open System Interconnection (OSI) model

Computer

7. Прикладной (Application)

6. Представления (Presentation)

5. Сеансовый (Session)

4. Транспортный (Transport)

3. Сетевой (Network)

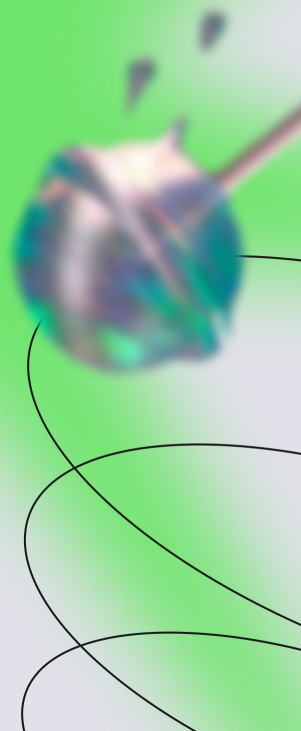
2. Канальный (Data Link)

1. Физический (Physical)

Network media







Протоколы прикладного уровня (Application)

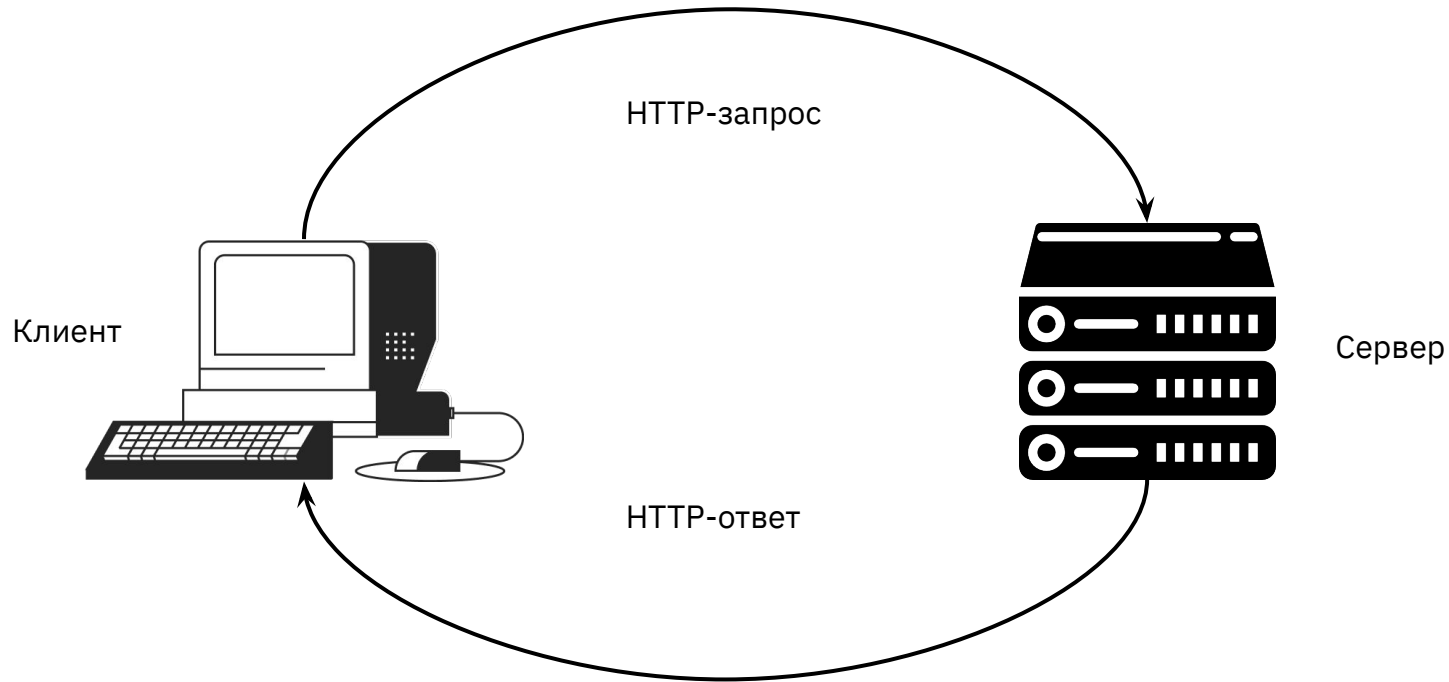




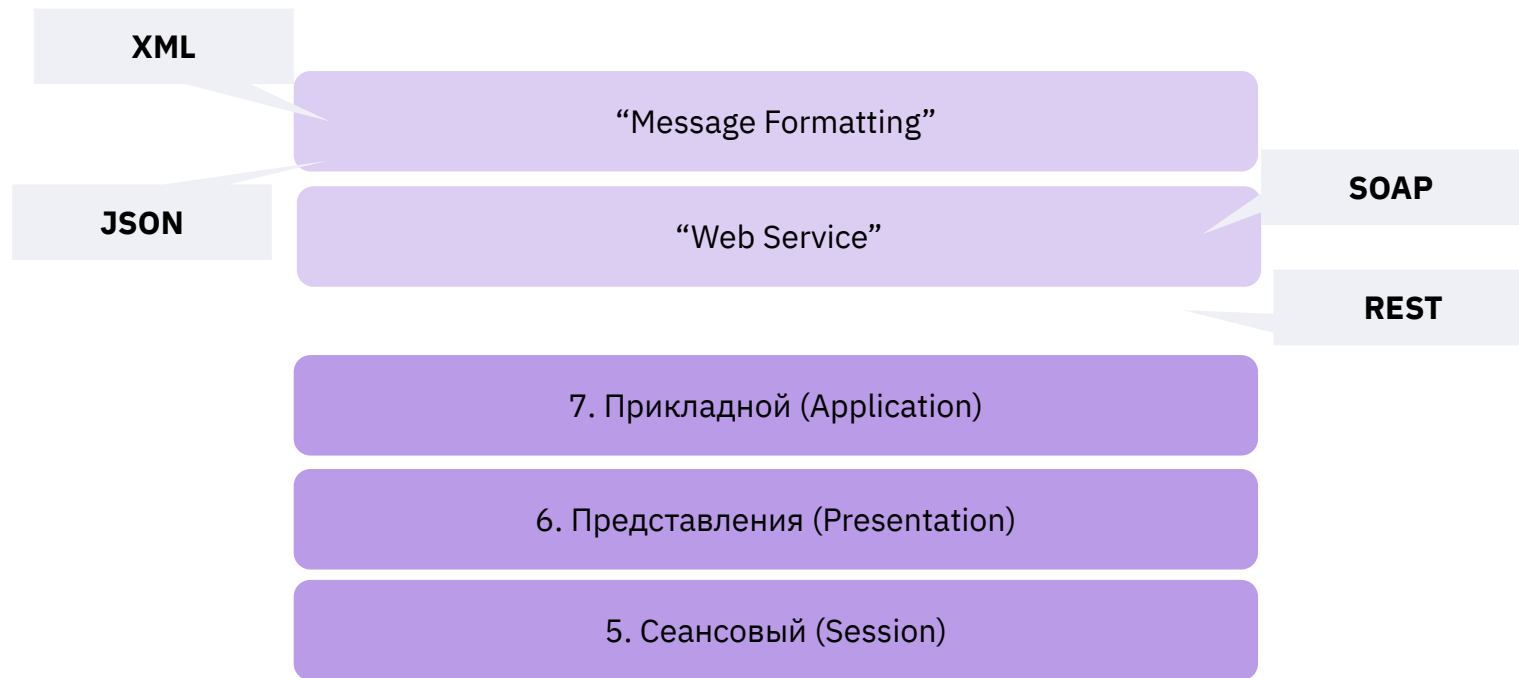
Протоколы прикладного уровня

-  HTTP (протокол передачи гипертекста)
-  FTP (протокол передачи файлов)
-  SMTP (простой протокол передачи почты)
-  DNS (Система доменных имен)

HTTP (Hypertext Transfer Protocol)



Open System Interconnection (OSI) model



SOAP vs REST

	SOAP	REST
Суть	Протокол	Архитектурный стиль
Состояние	Stateful, Stateless	Stateless
Формат	XML	XML, JSON, HTML, текст
Протокол передачи	HTTP, FTP, TCP, SMTP	HTTP
Скорость	медленный	быстрый
Кривая обучения	легко	сложно

XML vs JSON

```
<?xml version="1.0" encoding="UTF-8" ?>
  <dataset>
    <record>
      <id>1</id>
      <first_name>Kyle</first_name>
      <last_name>Danzey</last_name>

      <email>kdanzey0@dedecms.com</email>
    </record>
    <record>
      <id>2</id>
      <first_name>Stanly</first_name>
      <last_name>Chaise</last_name>
      <email>schaise1@php.net</email>
    </record>
    <record>
      <id>3</id>
      <first_name>Valentine</first_name>
      <last_name>Vasler</last_name>
      <email>vvasler2@ifeng.com</email>
    </record>
    <record>
      <id>4</id>
      <first_name>Herve</first_name>
      <last_name>Tollet</last_name>
```

```
{
  "dataset": {
    "record": [
      {
        "id": "1",
        "first_name": "Kyle",
        "last_name": "Danzey",
        "email": "kdanzey0@dedecms.com"
      },
      {
        "id": "2",
        "first_name": "Stanly",
        "last_name": "Chaise",
        "email": "schaise1@php.net"
      },
      {
        "id": "3",
        "first_name": "Valentine",
        "last_name": "Vasler",
        "email": "vvasler2@ifeng.com"
      },
      {
        "id": "4",
        "first_name": "Herve",
        "last_name": "Tollet",
        "email": "htollet3@chronoengine.com"
      }
    ]
  }
}
```

XML vs JSON

	XML	JSON
Читабельность	сложнее (язык разметки)	очень легко
Компактность кода	больше кода	меньше кода
Скорость парсинга	медленнее	быстрее
Простота синтаксиса	требует знания тегов	легко
Гибкость	у данных нет типа	работает с ограниченным количеством типов данных
Поддержка массивов	нет	да



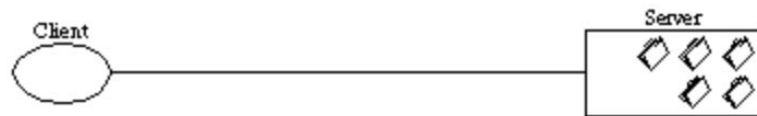
Representational State Transfer (REST)





Формирование REST

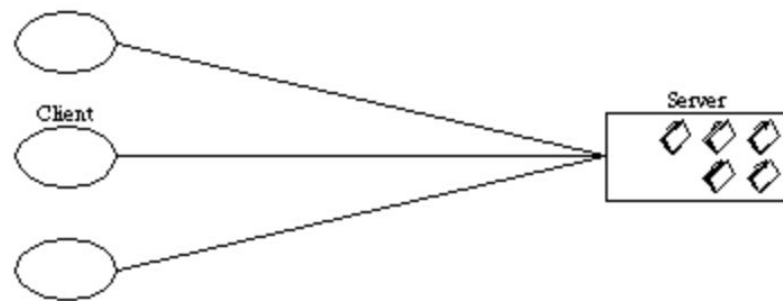
- **Разделение клиента и сервера**
- Stateless
- Cacheable
- Единый интерфейс
- Многослойная система
- Код по требованию





Формирование REST

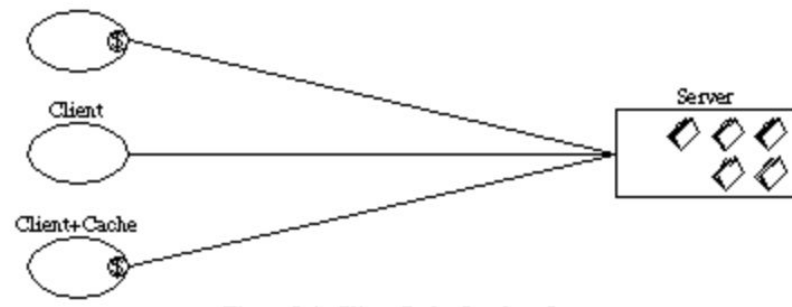
- Разделение клиента и сервера
- **Stateless**
- Cacheable
- Единый интерфейс
- Многослойная система
- Код по требованию





Формирование REST

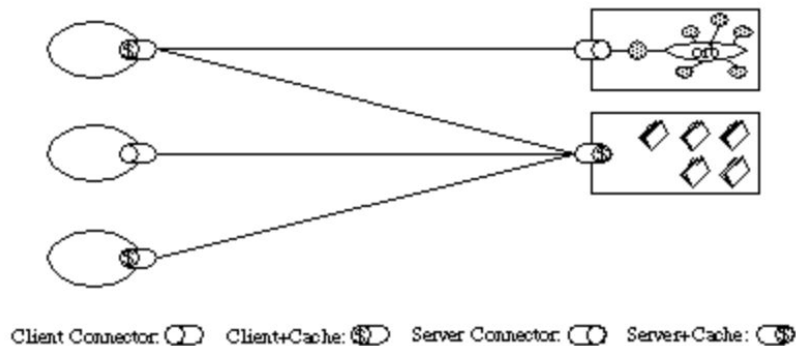
- Разделение клиента и сервера
- Stateless
- **Cacheable**
- Единый интерфейс
- Многослойная система
- Код по требованию





Формирование REST

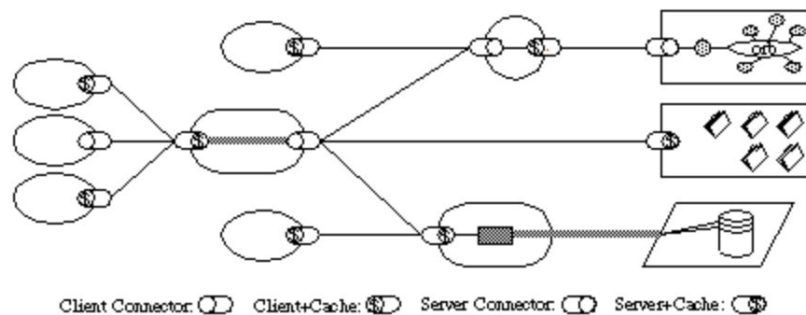
- Разделение клиента и сервера
- Stateless
- Cacheable
- **Единый интерфейс**
- Многослойная система
- Код по требованию





Формирование REST

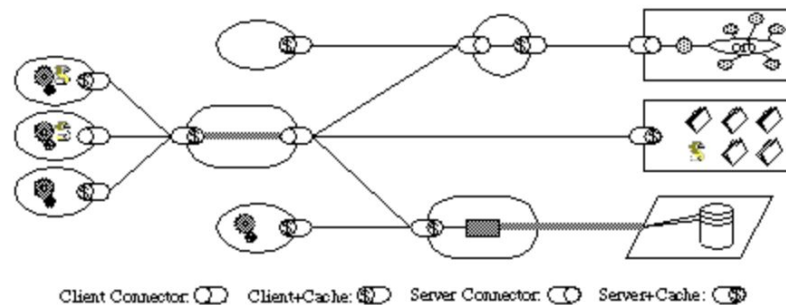
- Разделение клиента и сервера
- Stateless
- Cacheable
- Единый интерфейс
- **Многослойная система**
- Код по требованию





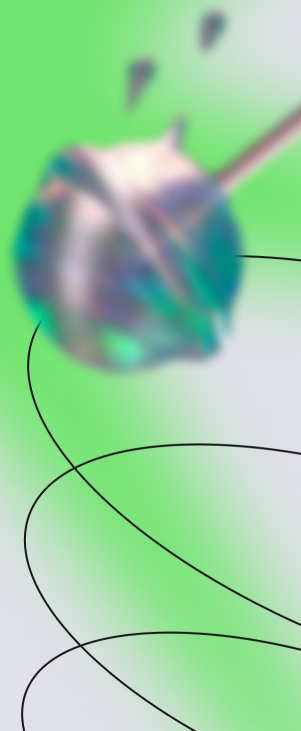
Формирование REST

- Разделение клиента и сервера
- Stateless
- Cacheable
- Единый интерфейс
- Многослойная система
- **Код по требованию**





Клиент-серверный поток HTTP





HTTP Request

```
1 POST /cgi-bin/process.cgi HTTP/1.1      ← Request Line
2 User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
3 Host: www.tutorialspoint.com
4 Content-Type: application/x-www-form-urlencoded
5 Content-Length: length
6 Accept-Language: en-us
7 Accept-Encoding: gzip, deflate
8 Connection: Keep-Alive
9
10 licenseID=string&content=string&/paramsXML=string
```

Request Header

← Empty Line

← Body



Методы HTTP



GET



POST



PUT



DELETE



HTTP Response

HTTP/1.1 200 OK	Status Line	HTTP Response
Date: Thu, 20 May 2004 21:12:58 GMT	General Headers	
Connection: close		
Server: Apache/1.3.27	Response Headers	
Accept-Ranges: bytes		
Content-Type: text/html	Entity Headers	
Content-Length: 170		
Last-Modified: Tue, 18 May 2004 10:14:49 GMT		
<p><html> <head> <title>Welcome to the Amazing Site!</title> </head> <body> <p>This site is under construction. Please come back later. Sorry!</p> </body> </html></p>		Message Body



Коды состояния

“200 OK”

“404 Not Found”

“403 Forbidden”

“500 Internal Server Error”



Работа в Postman





Заключение

- 💡 Понимание взаимодействия клиент-сервер и анализа API имеет важное значение для эффективного сбора, управления и обмена данными.
- 💡 Цикл "запрос-ответ" является основой взаимодействия клиент-сервер.
- 💡 REST — популярный архитектурный стиль для создания API благодаря своей простоте и масштабируемости.
- 💡 Модель Open System Interconnection (OSI) обеспечивает полезную основу для понимания различных уровней связи, участвующих во взаимодействии клиент-сервер.
- 💡 Postman, может значительно помочь дата-инженерам в создании и анализе HTTP-запросов и ответов, повышая производительность и эффективность сбора и обмена данными.
- 💡 Выбор между JSON и XML в качестве формата ответа.
- 💡 Использование Python и его популярных библиотек, таких как requests, значительно упрощает процесс выполнения HTTP-запросов и работы с API.