REST API

Michał Tęczyński
michal.teczynski@gmail.com

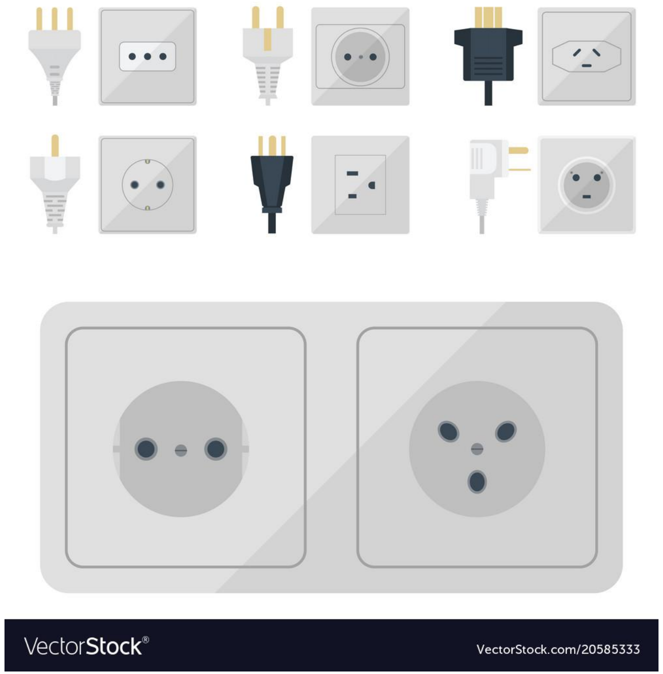
Introduction
Michał Tęczyński

- JS and PHP developer at Home.pl
- Studied at ZUT in Szczecin
- Email: michal.teczynski@gmail.com
- Github: https://github.com/michalv8
- LinkedIn: https://www.linkedin.com/in/michalteczynski/

What is API?

Set of clearly defined methods of communication between various software components.

What is API?


VectorStock®
VectorStock.com/20585333

USB TYPE C
USB TYPE A
USB MINI B
USB MICRO B
VectorStock®
VectorStock.com/19410930

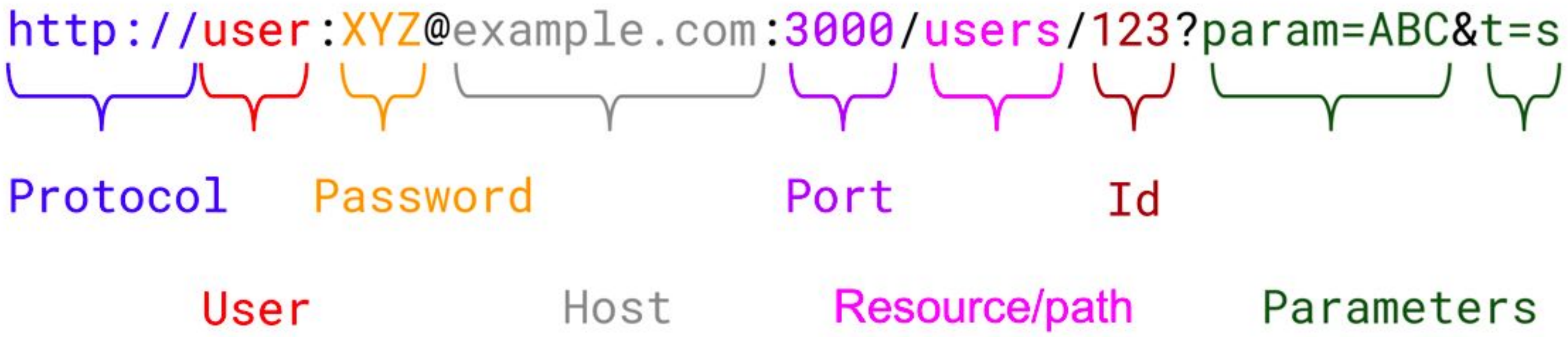
What is WEB API?

An **A**pplication **P**rogramming **I**nterface for either a web server or a web browser.

URL structure

Uniform Resource Locator

URL structure
Uniform Resource Locator



URL structure
Uniform Resource Locator

Required

- Protocol (https://)
- Host (example.com)
- Resource (/users)

Optional

- Authentication (user:ZYZ)
- Query (?param=ABC&t=s)
- Port (:3000)

Task #1

Install Chrome Advanced Rest Client

Task #1
Install Chrome Advanced Rest Client

- https://goo.gl/9uBc2A

Task #2

Make few simple REST requests

Task #2
Make few simple REST requests

- HOST: http://api.openweathermap.org
- Method: GET
- Resource: /data/2.5/weather
- Query
  - q: Szczecin
  - APPID: 0b3d75e5a49f2a267f054a0a60bed6f3
- Result:

Task #2
Make few simple REST requests

- HOST: http://api.openweathermap.org
- Method: GET
- Resource: /data/2.5/weather
- Query
  - q: Szczecin
  - APPID: 0b3d75e5a49f2a267f054a0a60bed6f3
- Result:
  - http://api.openweathermap.org/data/2.5/weather?q=Szczecin&APPID=0b3d75e5a49f2a267f054a0a60bed6f3

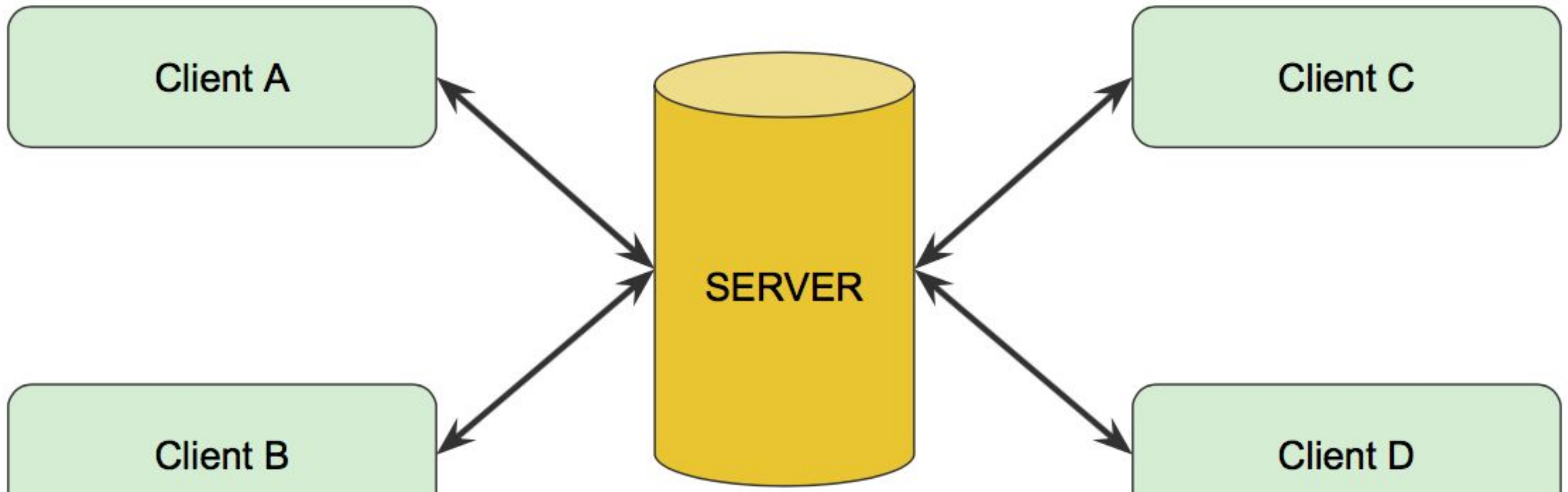
What is REST API?

Representational State Transfer (REST) is an architectural style that defines a set of constraints and properties based on HTTP

What is REST API?

- Client–server architecture
- Statelessness
- Uniform interface
- Hypermedia as the engine of application state (HATEOAS)

What is REST API?
Client-server architecture



What is REST API?
Statelessness

- Each request from client to server must contain all of the information necessary to understand the request, and cannot take advantage of any stored context on the server.
- Session state is therefore kept entirely on the client

What is REST API?
Uniform interface

- Resource identification in requests
- Resource manipulation through representations
- Self-descriptive messages
- Hypermedia as the engine of application state

What is REST API?
Uniform interface

Resource identification in requests

GET	/users
GET	/users/123
GET	/users/123/posts
POST	/users/123/posts
POST	/users/123/posts/456/comments
PUT	/users/123/posts/456/comments/789
DELETE	/users/123/posts/456/comments/789

What is REST API?
Uniform interface

Resource manipulation through representations

```
GET /users/123

{
      "id": 123,
      "name": "Chuck Norris",
      "city": "New York"
}
```

```
PUT /users/123

{
      "id": 123,
      "name": "Chuck Norris",
      "city": "Szczecin"
}
```

What is REST API?
Uniform interface

Self-descriptive messages

- Each message includes enough information to describe how to process the message.

What is REST API?
Uniform interface

Hypermedia as the engine of application state (HATEOS)

```
GET /api/users/123

{
     "id": 123,
     "name": "Chuck Norris",
     "city": "New York",
     "_links": {
          "_self": "https://example.com/api/users/123",
          "posts": "https://example.com/api/users/123/posts",
          "comments": "https://example.com/api/users/123/comments"
     }
}
```

What is REST API?
Versioning

- Backward compatibility when backend is under active development
- Can be a part of URL
  - **GET /api/v2/users/123**
- Or as a header

```
GET /api/users/123
Accept: application/json; version=2.4
```

What is REST API?
Status codes

- Describing what has happened to resource
- Clients should check them

What is REST API?
Status codes

Method	Status codes
GET	200, 400, 5xx
POST	201, 400, 404, 5xx
PUT	204, 400, 404, 5xx
PATCH	204, 400, 404, 5xx
DELETE	204, 404, 5xx

Authentication

Authentication
Common methods

- Basic auth
- Api key
- OAuth (1/2)

Authentication
Basic auth

- Simple user and password data
- Used mostly in backend-to-backend communication
- Sent as base 64 encoded string in request headers

```
GET /api/users/123
Host: example.com
Authorization: Basic QWxhZGRpbjpPcGVuU2VzYW1l
```

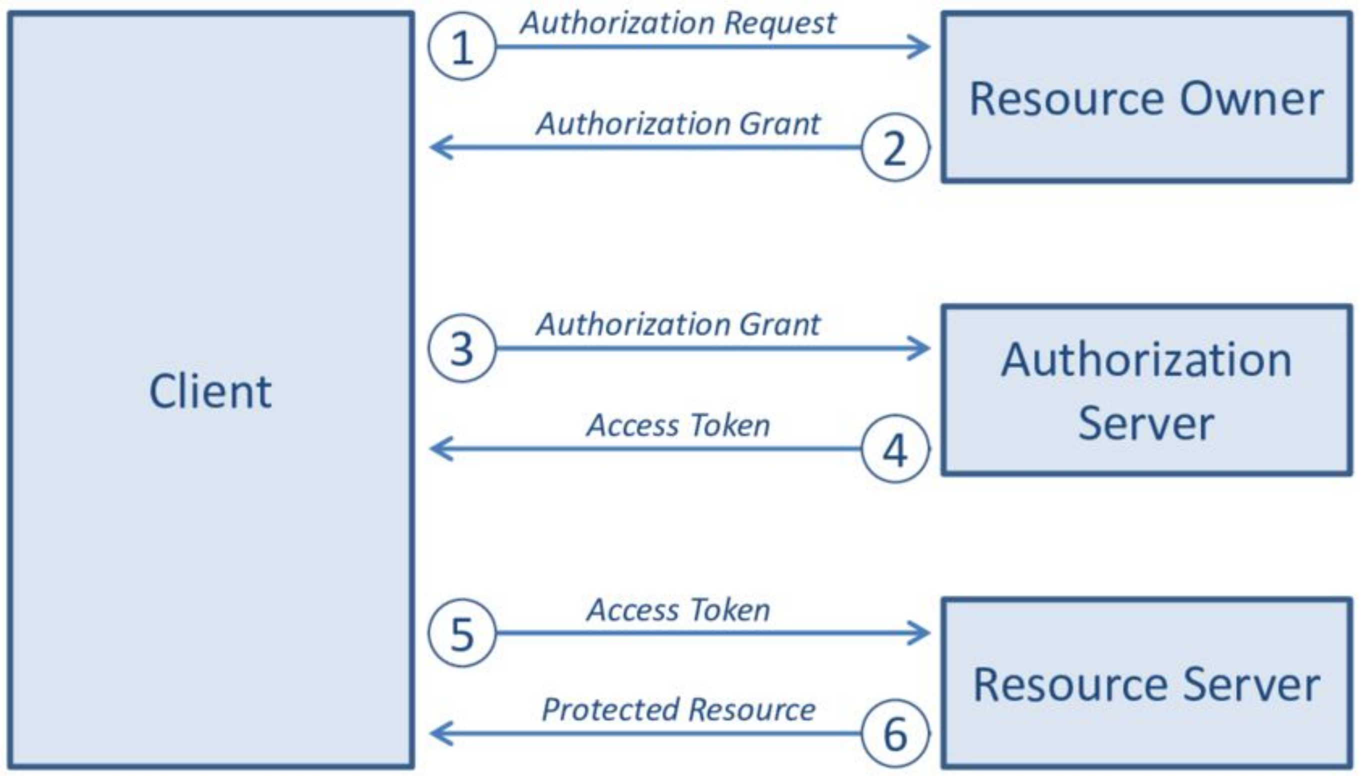
Authentication
Api key

- Sign is generated on the server side
- Client is signing requests by this value
- Sent usually as query parameter or custom header

```
GET /api/users/123?apikey=QWxhZGRpbjpPcGVuU2VzYW1l
Host: example.com
```

```
GET /api/users/123
Host: example.com
Api-Key: QWxhZGRpbjpPcGVuU2VzYW1l
```

Authentication
OAuth2



JSON

Java**S**cript **O**bject **N**otation

JSON
Javascript Object Notation

- Simpler and more readable than its main “contender” - XML
- Has enough complexity to fulfil REST needs
- Is widely used in REST APIs
- Can be put straight through JS code

JSON
Structure

```
{
      "id": 123,
      "name": "John Doe",
      "middle_name": null,
      "balance": 456.45,
      "active": true,
      "address": {
            "city": "Szczecin",
            "street": "Cyfrowa 6"
      },
      "transactions": [
            {
                  "amount": -123.45
            },
      ],
}
```

Request methods

GET, POST, PUT, PATCH, DELETE, OPTIONS

Request methods
GET

- Used for retrieving particular resource or list of resources
- No body
- Can be combined with query parameters

```
GET /api/users
GET /api/users/123
```

Request methods
POST

- Used for creating new resources
- Has body
- Can be combined with query parameters

```
POST /api/users
{
    "name": "Luke Skywalker",
    "planet": "Tatooine"
}
```

Request methods
PUT

- Used for modifying existing resources
- Identifying resource to edit by its unique ID
- Has body
- Whole resource is modified

```
PUT /api/users/123
{
    "id": 123,
    "name": "Luke Skywalker",
    "planet": "Dagobah"
}
```

Request methods
PATCH

- Used for modifying properties of existing resources
- Identifying resource to edit by its unique ID
- Has body
- Only properties existing in body are modified

```
PATCH /api/users/123
{
    "planet": "Ahch-To"
}
```

Request methods
DELETE

- Used for deleting particular resource
- Resource is identified by its unique ID
- Request with no body

```
DELETE /api/users/123
```

Request methods
Idempotent and safe methods

- **Safe methods** does not change resources
- **Safe methods** are cachable
- **Idempotent methods** can change resource
- **Idempotent methods** can be called with the same arguments many times without repercussions

Request methods
Idempotent and safe methods

Method	Idempotent	Safe
GET	Yes	Yes
POST	No	No
PUT	Yes	No
PATCH	No	No
DELETE	Yes	No

Task #3

Make more advanced REST requests

REST API
REST API EVERYWHERE
frabz.c
memegenerator.net

Task #3

Make more advanced REST requests

- https://coinmarketcap.com/api/
- https://exchangeratesapi.io/
- https://jobs.github.com/api
- Public API list:
  - https://github.com/toddmotto/public-apis

Task #3
https://coinmarketcap.com/api/

- Get list of all cryptocurrencies
- Get list of ticker data limited to 10 entries and sorted by rank
- Get ticker data of Polcoin (PLC) with prices in PLN

Task #3

https://coinmarketcap.com/api/

- Get list of all cryptocurrencies
  - **https://api.coinmarketcap.com/v2/listings/**
- Get list of ticker data limited to 10 entries and sorted by rank
  - **https://api.coinmarketcap.com/v2/ticker/?limit=10&sort=rank**
- Get ticker data of Polcoin (PLC) with prices in PLN
  - **https://api.coinmarketcap.com/v2/ticker/257/?convert=PLN**

Task #3

https://exchangeratesapi.io/

- Get list of latest exchange rates for PLN
- Get list of exchange rates at 2008-06-08 for USD limited to only PLN, USD and EUR

Task #3
https://exchangeratesapi.io/

- Get list of latest exchange rates for PLN
    - **https://exchangeratesapi.io/api/latest?base=PLN**

- Get list of exchange rates at 2008-06-08 for USD limited to only PLN, USD and EUR
    - **https://exchangeratesapi.io/api/2008-06-08?symbols=USD,GBP**

Task #3

https://jobs.github.com/api

- Find full time job positions for C++ programmer in Barcelona

Task #3
https://jobs.github.com/api

- Find full time job positions for C++ programmer in Barcelona
  - **https://jobs.github.com/positions.json?description=c++&location=Barcelona&full_time=true**

Firebase

https://firebase.google.com/

EVERYTHING IS IN
"THE CLOUD"
makeameme.org

Firebase
What is this?

- Serverless (cloud) service architecture
- Lots of services which can be accessed by their Web APIs or programming libraries
- Main (free) services
  - Hosting
  - Database
  - Authorization

Firebase
Realtime database

- Located in Google Cloud
- NoSQL (not relational) structure (basically it’s just a big JSON file)
- Quite advanced permissions settings
- Can be accessed in real time by multiple clients
- Can also be accessed through REST API
  - https://firebase.google.com/docs/reference/rest/database/

Task #4

Create Google account and open Firebase Console

Task #4

Create Google account and open Firebase Console

- Create Google account - https://goo.gl/5Fstza
- Open Firebase console - https://console.firebase.google.com

Task #5

Create team projects and setup permissions

Task #6

Setup database permissions

Task #6
Setup database permissions

- Set full read and write permission for learning purposes
- **!!! Never do it on live, production projects !!!**

```
{
    "rules": {
        ".read": true,
        ".write": true
    }
}
```

Fetch API

Fetch API

- Native API, available in modern browsers (https://caniuse.com/#search=fetch) to provide ability to make HTTP requests
- Low-level API
- Based on promises
- Official docs - https://goo.gl/vNPYZA
- Polyfill for older browsers - https://github.com/github/fetch

Fetch API

```
fetch('https://api.ipify.org?format=json')
    .then(function(response) {
        return response.json();
    })
    .then(function(myJson) {
        console.log(myJson);
    });
```

Fetch API

```
fetch('https://jsonplaceholder.typicode.com/posts', {
       method: 'POST',
       body: JSON.stringify({
           title: 'foo',
           body: 'bar'
       }),
       headers: {
           "Content-type": "application/json; charset=UTF-8"
       }
   })
   .then(function(response) {
       return response.json();
   })
   .then(function(json) {
       console.log(json);
   });
```

JS Tasks

Thank you