BULZAN STEFAN

WEB APPLICATIONS

A short presentation

WEB APPLICATIONS

- A web application is an application that transmits data over the network
- To transport data somewhere you need to establish the following
  - WHERE do you want to send the data
  - HOW do you send the data there
  - WHAT do you send, WHAT format do you use
- WWW (World Wide Web) uses mainly the following technologies to respond to these questions:
  - WHERE? : TCP/IP
  - HOW? : HTTP/HTTPS
  - WHAT? : HTML, CSS, SCRIPTS

WHERE? TCP/IP

- TCP : Transmission Control Protocol
- IP : Internet Protocol
  - Organises the data in packages
  - Ensures that the data is received correctly
  - Ensures that the data can be reconstructed correctly
- Each node in the network is assigned an unique reference (IP address)

HOW? HTTP !

- HTTP : HyperText Transfer Protocol
- Establishes a communication protocol between the client(browser) and server
- It’s a request/response protocol. In order to receive data you need to request it
- HTTP components
  - request
    - URL: location of the data: https://recruit-me.it/moodle/my/
      - <protocol>://host[:port]/path/to/resource
    - URL parameters: extra information for the url
      - /resource?query=value&key=another
      - As you see, the url params are key, value pairs that are separated by ? In the start and then &
    - Headers: extra information about the request (key-value)
    - Http verb: defines the intention: GET, POST, PUT, PATCH, DELETE
    - Body: the data payload

HOW? HTTP RESPONSE

- Response
  - body : the response
  - code : a special code that defines the type of the response:
    - [100..199] - info
    - [200..299] - success. 200 is OK
    - [300-399] - redirects
    - [400-499] - client errors
    - [500..599] - server errors

WHAT? HTML

- WWW uses HTML (and CSS, JS) to present the data to the users
- HTML = HyperText Markup Language
- It’s an xml based format that defines a web page:
- <html> : defines a web page
- <head> : the head part of the page, contains metadata, links
- <body> : the body of the page that will be shown
- Other format tags: p, h*, div, span, label, button, table, tr, th

THIS IS HOW WWW WORKS.
WHAT ABOUT SENDING RAW DATA?

APIS

- With the advent of multiple mediums of presenting the data, having a web application that serves data as HTML, CSS and JS became obsolete
- Now we have mobile devices that present data in different ways
- Also we want to have access to data directly without it being presented in a visual way
- These were some of the reasons why the presentation of data was decoupled from the data itself
- Now we have API services that only give us the data and then this data can be consumed by other services that produce the HTML needed or other formats as the client needs

REST

- REST = REpresentational State Transfer
- Inspired by WWW, REST took all the good parts from HTTP and brings them to the application world
- REST uses HTTP as it was designed:
  - A resource is defined by url
    - If I want the person with id 2: http://host:port/application/persons/2
  - We use HTTP verbs to describe the operation
    - If I want to get the information, I'll use GET
  - We can put headers to provide more information about the request
  - We can use the body to send information
    - To save new person, we send a POST at persons/ with the person as body

DATA FORMAT

- What data format do we use for REST. How would a person look in the REST body?
- You can use XML, but... we can do better
- Here comes JSON (JavaScript Object Notation)
- The format is simple:
  - {“key”:”value”}
  - And for arrays {“key”:”value”, “array”:[“val1”,”val2”]}
- Of course you can nest the objects

```
{
  “name” : “Maria”,
  “address” :{
    “street”:”Moldovei”,
    “number”:19
  },
  “grades”: [10,6,7,9]
}
```

REST URL

- There are some rules when it comes to build the urls to locate a resource
- It should ALWAYS contain ONLY nouns
- The operations should be described by the VERB, not by url
- If the resource is a collection, it should be PLURAL
  - Eg. We will have multiple persons, the url is /persons
- If I want to get all resources you have a GET on /persons
- If you want to filter or sort the resources you can specify url params:
  - /persons?sortBy=name&name=Popescu
- If you want a single resource, you specify its id: GET on /persons/2