APIS, REST & HTTP

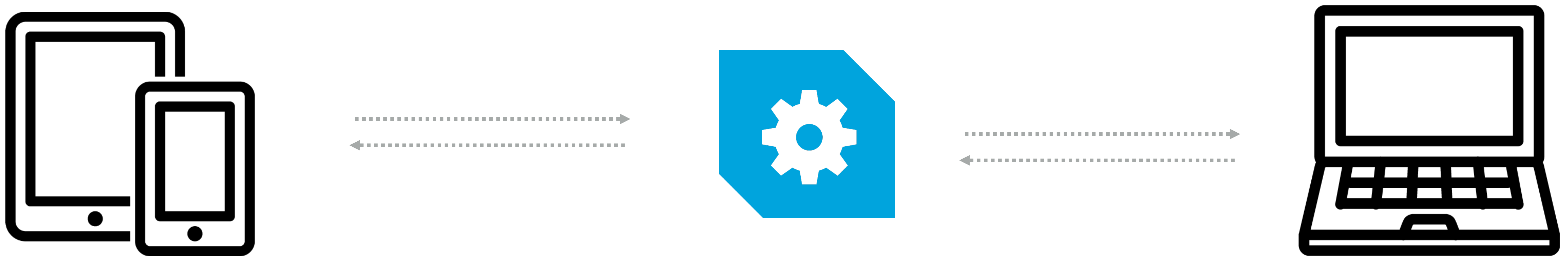
BACKEND-WORKSHOP

Michael Fröhlich - michael-froehlich@cdtm.de

Tobias Dümmling - tobias.duemmling@cdtm.de

WHAT IS AN API?

APPLICATION PROGRAMMING INTERFACE



The API is a set of function or procedure used by computer programs to access services from the operating system, software libraries or other services.

https://www.youtube.com/watch?v=s7wmiS2mSXY

SO, HOW DO APIS WORK?

- UK Police Forces https://data.police.uk/docs/
  - Getting all the different police forces: https://data.police.uk/docs/forces
  - Getting a specific police force: https://data.police.uk/docs/forces/city-of-london
  - …
- In short: *APIs return machine readable data.*

REST

REPRESENTATIONAL STATE TRANSFER

70% of public APIs now rely on REST services

REST is *an architecture style* for designing networking services and APIs. The idea is that, by having universal constraints on how APIs should look, it is easier to maintain and use them.

Important: REST is not an actual protocol, but rather a set of constraints.

REST

THEORY

A set of constraints? What?

1. Uniform Interface
    1. **Resource-Based:** Individual resources are identified in requests using URIs. The resources themselves are conceptually separate from the representations that are returned to the client.
    2. **Manipulation of Resources Through Representations:** When a client holds a representation of a resource, he can modify it, given he has the required permissions
    3. **Self-descriptive Messages:** Each message includes enough information to describe how to process the message.
    4. **Hypermedia as the Engine of Application State:** Where necessary, links are contained in the returned body
2. Stateless
    - All the necessary information to handle the request is contained within the request itself,
3. Cacheable
    - Everything on the web can be cached. Responses must, implicitly or explicitly, define themselves as cacheable, or not, to prevent clients reusing stale or inappropriate data in response to further requests.
4. Client-Server
    - The uniform interface separates server and client. They may also be replaced and developed independently, as long as the interface is not altered.
5. Layered System
    - A client cannot ordinarily tell whether it is connected directly to the end server, or to an intermediary along the way. (improves scalability)
6. Code on Demand (optional)
    - Servers are able to temporarily extend or customize the functionality of a client by transferring logic to it that it can execute. Examples of this may include compiled components such as Java applets and client-side scripts such as JavaScript.

Source: http://www.restapitutorial.com/lessons/whatisrest.html

7 GOLDEN REST ROUTES

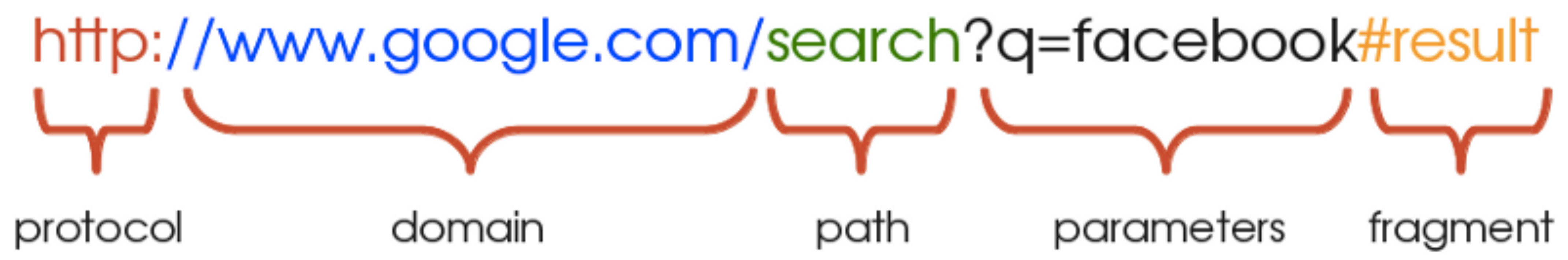
- All resources are referred to as noun. The HTTP method defines the action.
- REST almost always uses HTTP as underlying protocol

Action	URL	HTTP Verb	Description
index	/photos/	GET	Get a list of all photos.
new	/photos/new	GET	Get the necessary information to let the user create a new photo.
create	/photos	POST	Create a new photo resource on the server.
show	/photos/:id	GET	Get the photo with the specified id.
edit	/photos/:id/edit	GET	Get the necessary information for the user to be able to edit the photo.
update	/photos/:id	PATCH/PUT	Update an existing photo.
destroy	/photos/:id	DELETE	Delete the photo with the specified id.

Example Github issues: https://github.com/

HTTP

- HTTP stands for **Hypertext Transfer Protocol**
- It's the network protocol used to deliver virtually all files and other data (collectively called *resources*) on the World Wide Web
- HTTP is used to transmit *resources*, not just files. A resource is some chunk of information that can be identified by a URL (Uniform Resource Locator)



Interesting Read: The difference between URI and URL https://danielmiessler.com/study/url-uri/#gs.TfuRvHk

HTTP METHODS

- HTTP Methods tell a server how to process a request

Method	Description
GET	The GET method is used to retrieve information from the given server using a given URI. Requests using GET should only retrieve data and should have no other effect on the data.
POST	A POST request is used to send data to the server, for example, customer information, file upload, etc. using HTML forms.
PUT	Replaces all current representations of the target resource with the uploaded content.
DELETE	Removes all current representations of the target resource given by a URI.
…	

HTTP Methods: https://www.tutorialspoint.com/http/http_methods.htm

HTTP REQUEST / RESPONSE

- Each HTTP Request / Response is built up of 2 parts
  1. Header
     - Contains general information about the request: content-type, status, …
  2. Body
     - Contains the actual content
- Each HTTP Response has a status code, which gives information about the success of the request.
  - 2xx -> success
  - 4xx -> client error
  - 5xx -> server error

Full list of HTTP status codes: http://www.restapitutorial.com/httpstatuscodes.html

HTTP BODY: JSON

JAVASCRIPT OBJECT NOTATION

- APIs widely use JSON to transfer data
- JSON is a syntax for storing and exchanging data.
  - JSON is a lightweight data-interchange format
  - JSON is language independent
  - JSON is "self-describing" and easy to understand

```
// Example JSON Object
{
    "Name": "John Doe",
    "PermissionToCall": true,
    "PhoneNumbers": [{
        "Location": "Home",
        "Number": "555-555-1234"
    }, {
        "Location": "Work",
        "Number": "555-555-9999 Ext. 123"
    }]
};
```

7 GOLDEN REST ROUTES

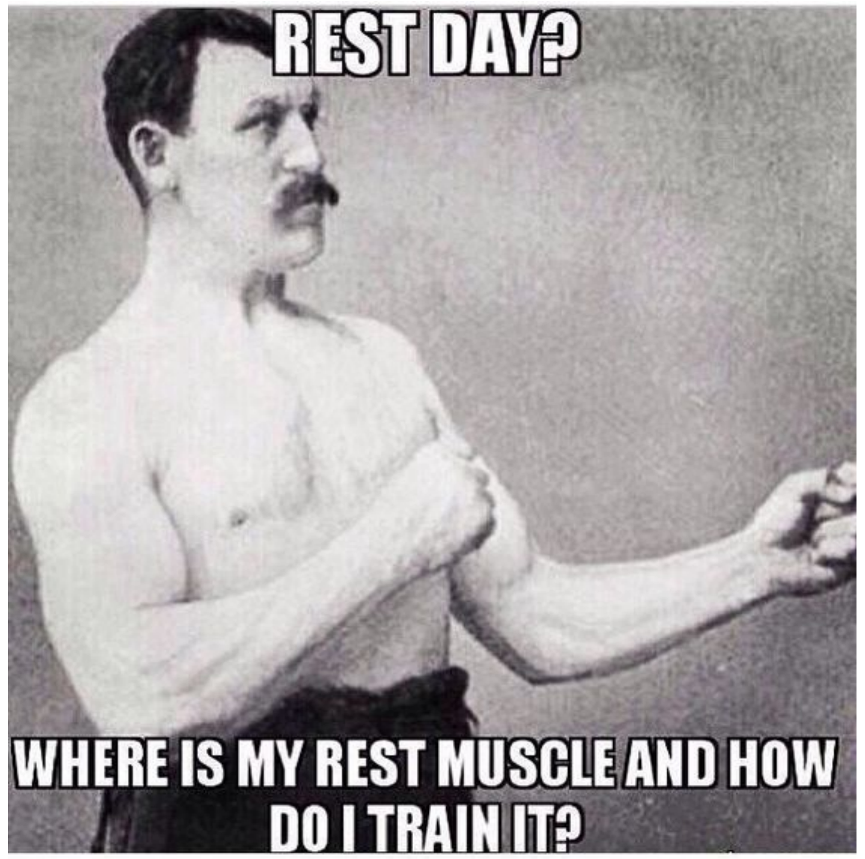
- All resources are referred to as noun. The HTTP method defines the action.
- REST almost always uses HTTP as underlying protocol

Action	URL	HTTP Verb	Description
index	/photos/	GET	Get a list of all photos.
new	/photos/new	GET	Get the necessary information to let the user create a new photo.
create	/photos	POST	Create a new photo resource on the server.
show	/photos/:id	GET	Get the photo with the specified id.
edit	/photos/:id/edit	GET	Get the necessary information for the user to be able to edit the photo.
update	/photos/:id	PATCH/PUT	Update an existing photo.
destroy	/photos/:id	DELETE	Delete the photo with the specified id.

Example Github issues: https://github.com/

CODING CHALLENGE

- Rest API
  - Implement the 'index' route for lists
  - Implement the 'index' route for tasks



Info: You can find a detailed exercise description in the repositories wiki.