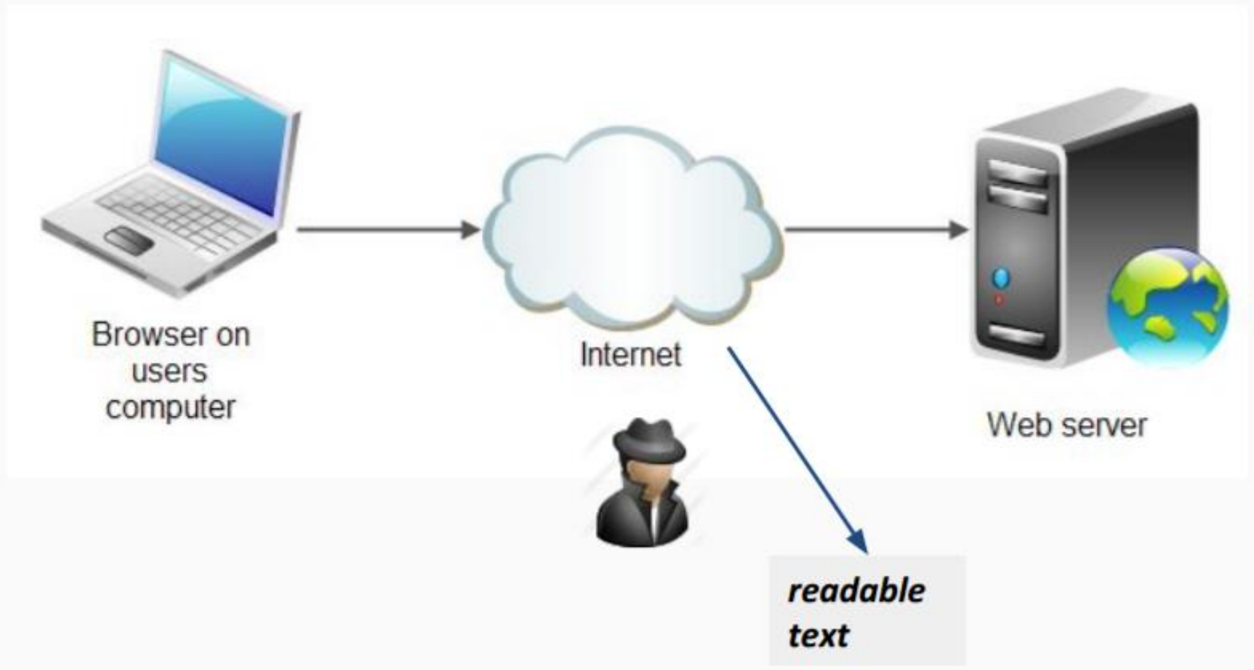
Școala informală de IT

Introduction to Web

HTTP

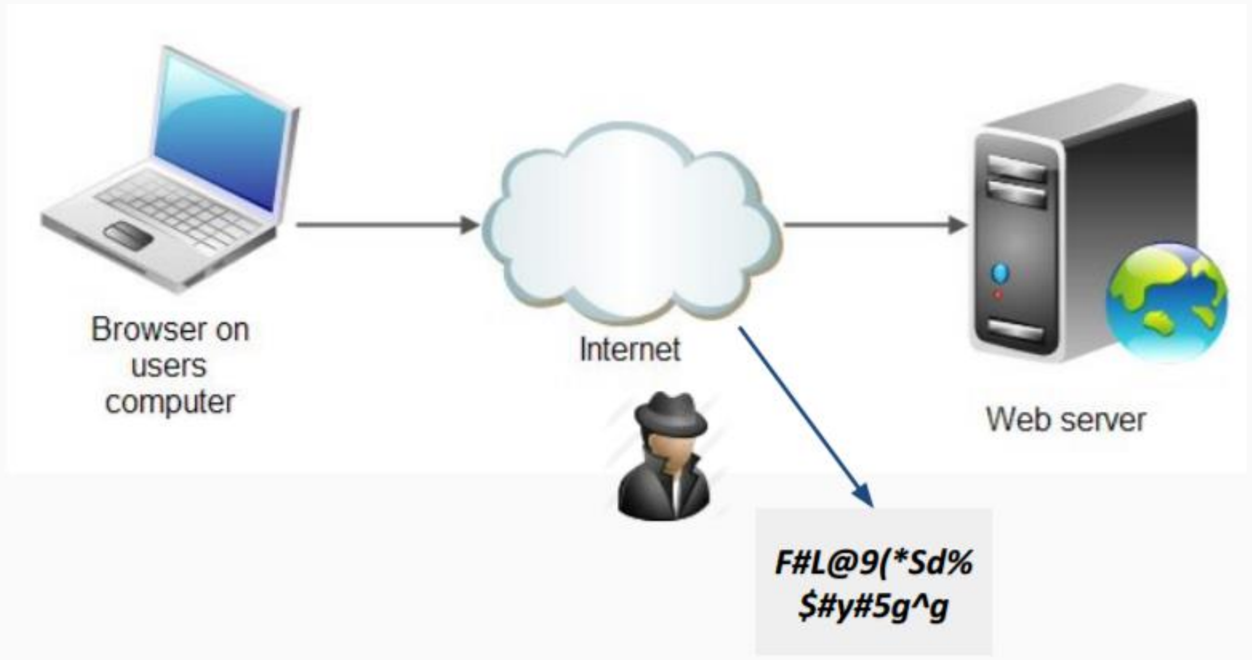
HTTP = Hyper Text Transfer Protocol
HTTP is designed to enable communications between clients and servers and works as a request-response protocol between a client and server



``

HTTPS

HTTPS = Hyper Text Transfer Protocol Secured

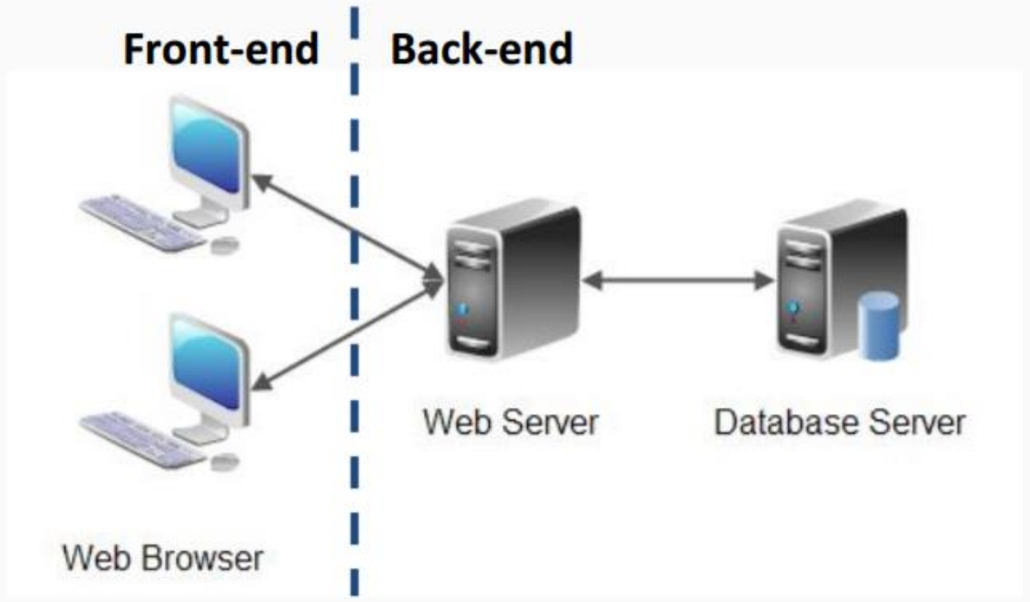


``

Web Application Architecture

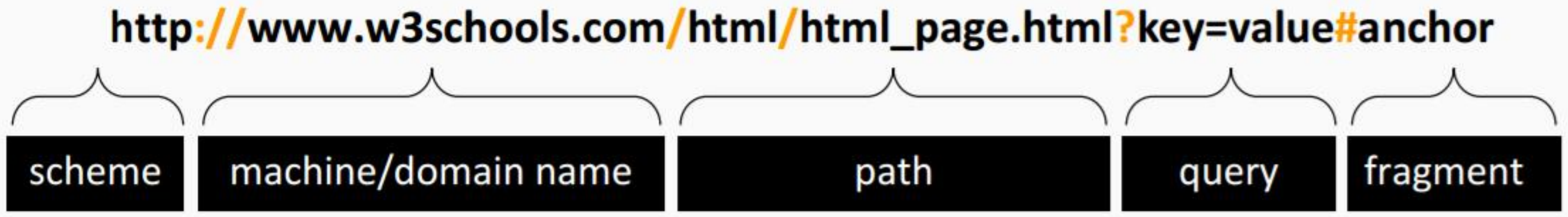
In the browser = Presentation layer
Web Server = Business logic layer
Database Server = Data layer

Front-end = Web Browser
Back-end = Web Server + Database Server



URL

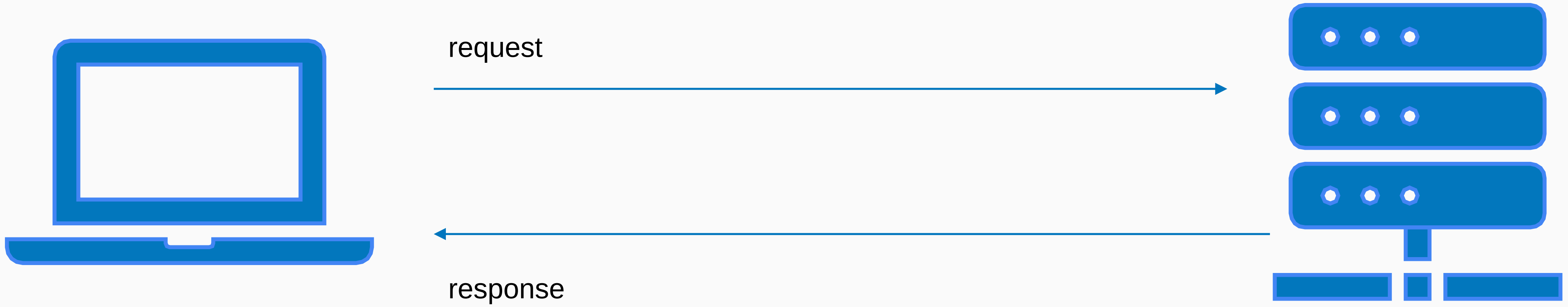
URL Example (explained) http://www.w3schools.com/html/html_page.html?key=value#anchor


http://www.w3schools.com/html/html_page.html?key=value#anchor
scheme
machine/domain name
path
query
fragment

1. **http** is the **scheme**
2. semicolon and two slashes (://) separate the scheme from the machine/domain name
3. **www.w3schools.com** is the **machine/domain name. www is subdomain.**
4. single slash (/) separates the name from the path
5. **html/html_page.html** is the **path**
6. question mark (?) separates the path from query
7. **key=value** is the query (which are key-value pairs separated by **&**. Ex.: key1=value1**&**key2=value2**&**...)
8. hash tag (#) separates the query from fragment
9. **anchor** is the fragment where to scroll the page when loading the url

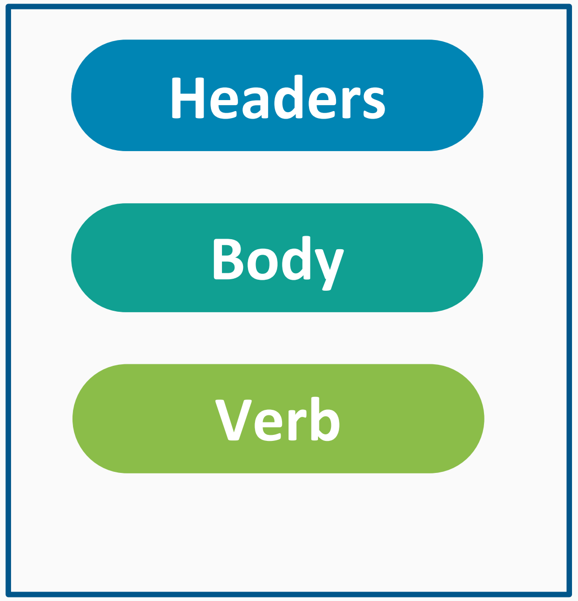
Web 101

Client-server architecture

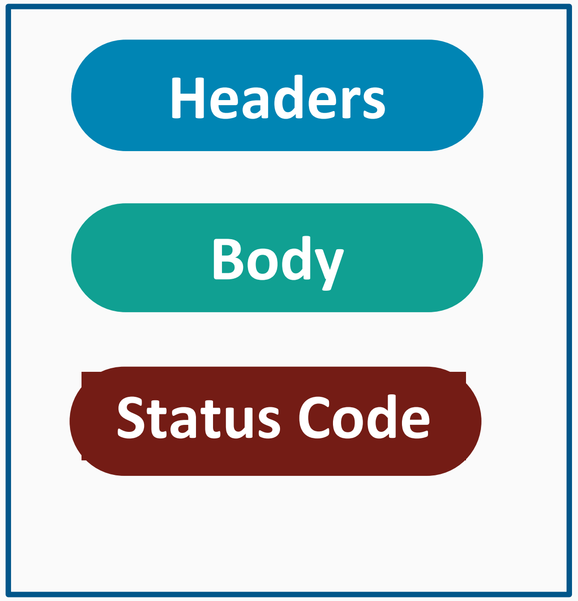


Request/Response

Request



Response



Request/Reponse

Request

- Headers
- Body
- Verb

Response

- Headers
- Body
- Status Code

HTTP Verbs

Verb

Verbs/HTTP methods	Description
GET	retrieves information from the given server using a given URI
POST	send data to the server, for example, customer information, file upload, etc. using HTML forms.
PUT	Replaces all the current representations of the target resource with the uploaded content.
DELETE	Removes all the current representations of the target resource given by URI.

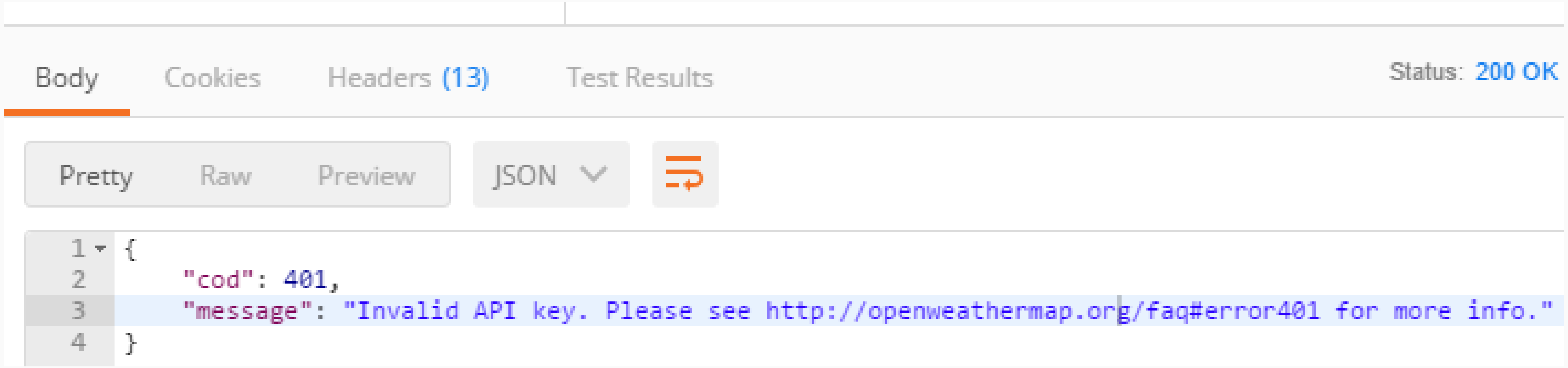
Status codes categories

Code	Description
1xx: Informational	request was received and the process is continuing.
2xx: Success	action was successfully received, understood, and accepted.
3xx: Redirection	further action must be taken in order to complete the request
4xx: Client Error	request contains incorrect syntax or cannot be fulfilled.
5xx: Server error	server failed to fulfill an apparently valid request.

Status Codes

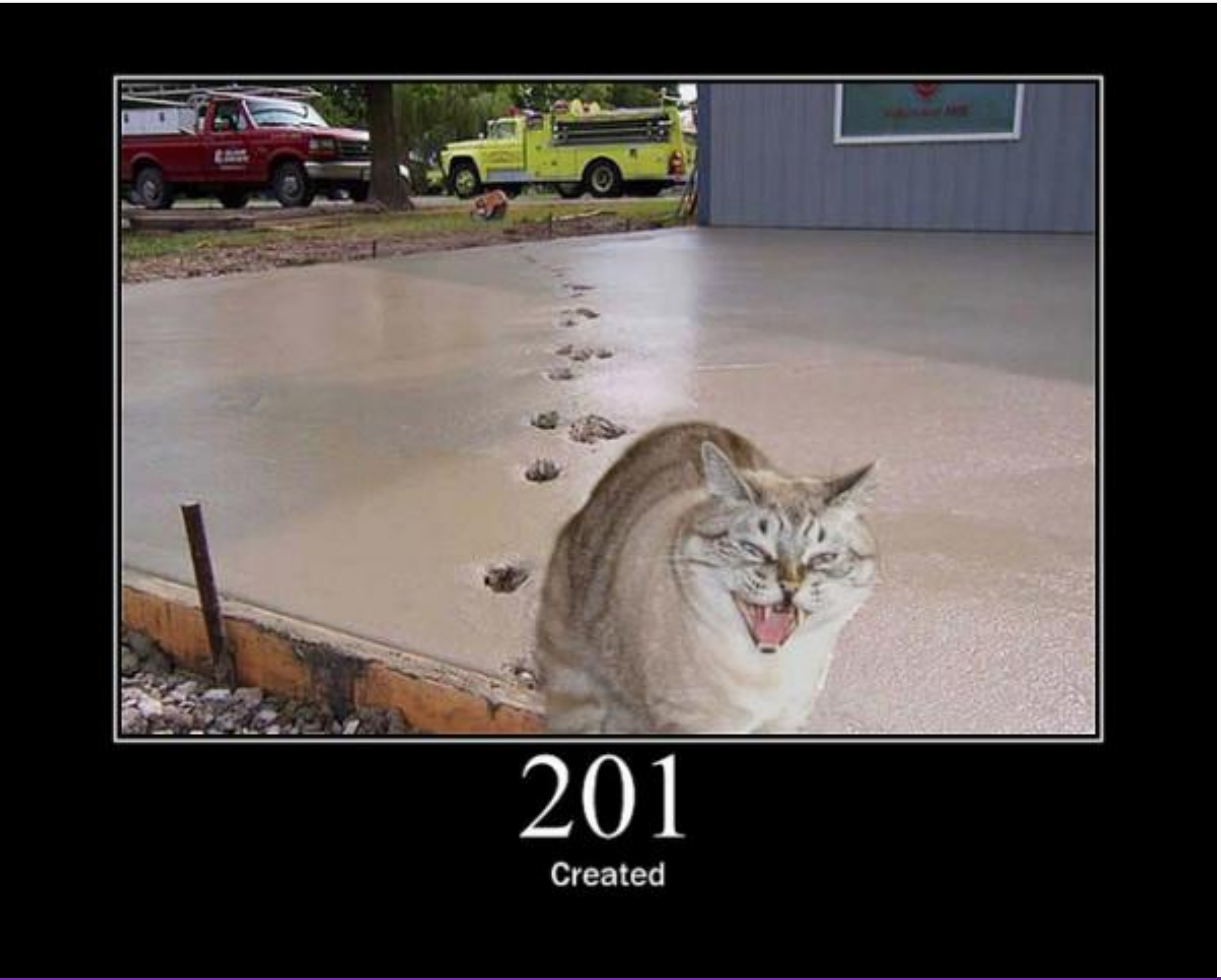
OK ?!

Or When OK really means OK



Status: 200 OK

```
{
    "cod": 401,
    "message": "Invalid API key. Please see http://openweathermap.org/faq#error401 for more info."
}
```


201
Created

ccepted

- For further processing
- For enqueuing something

- After deleting a resource

401

Unauthorized

403

- Identified but not allowed to perform an operation

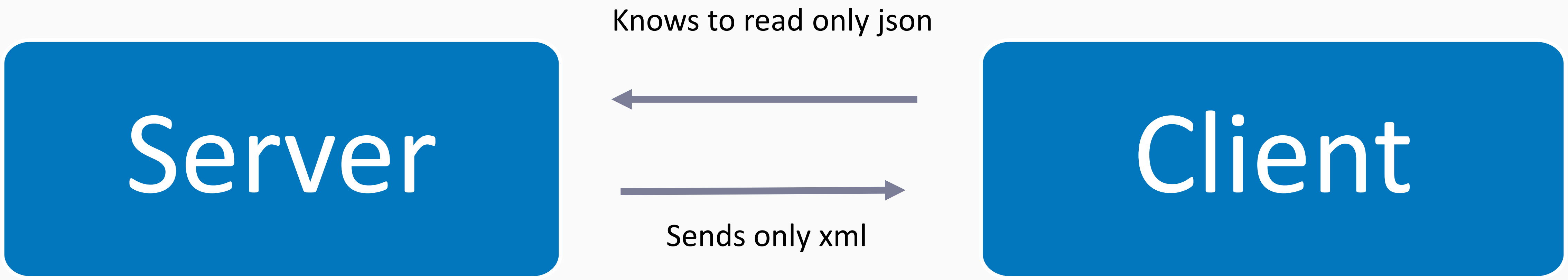
Inapropriate picture

404
Not Found

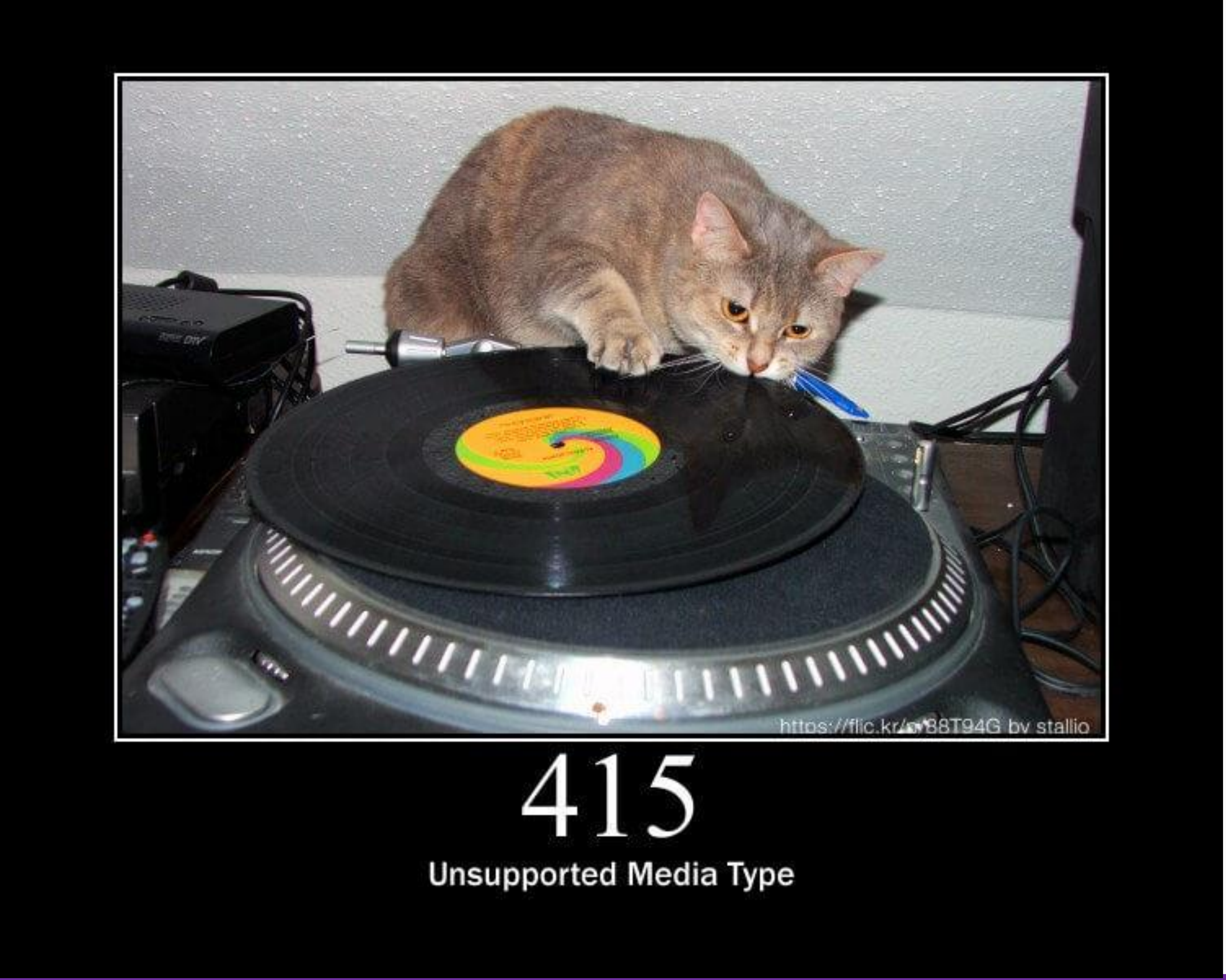
405
Method Not Allowed

406 – Not Acceptable

- Accept header – xml
- Server does only json



409
Conflict



Accept header has a media type you don’t support

Headers

Accept

- The mime type the client prefers and understands

Accept : text/xml
Accept : application/json

Content-Type

- The mime type of the request body
- Describes the content of the request

Content-Type : text/xml

Cache-Control

Cache-Control : no-cache
max-age=x(seconds)

Cache

- **Cache** stores recently used information so that it can be quickly accessed at a later time

- A web cache stores temporary copies of web pages to improve performance and reduce bandwidth consumption. At any given time, you may be browsing the web through several layers of caches.

- Types:
  - Client caches: store info on the client side(in the browser)
  - Server – store info in the memory of the server to prevent a round-trip to the database

Cookies

- Expiration Date
- Associated to a domain
- Name
- Value associated to that name

Cookies

Cookies

- Cookies provide another way you can store information for later use. Cookies are small files that are created on the client’s hard drive (or, if they’re temporary, in the web browser’s memory)

Advantages

- user doesn’t know what is stored in them
- easily used by any page in your application and even retained between visits, which allows for truly long-term storage

Disadvantages

- they’re limited to simple string information
- they’re easily accessible and readable if the user finds and opens the corresponding file
- poor choice for complex or private information or large amounts of data
- some users disable cookies on their browsers, which will cause problems for web applications

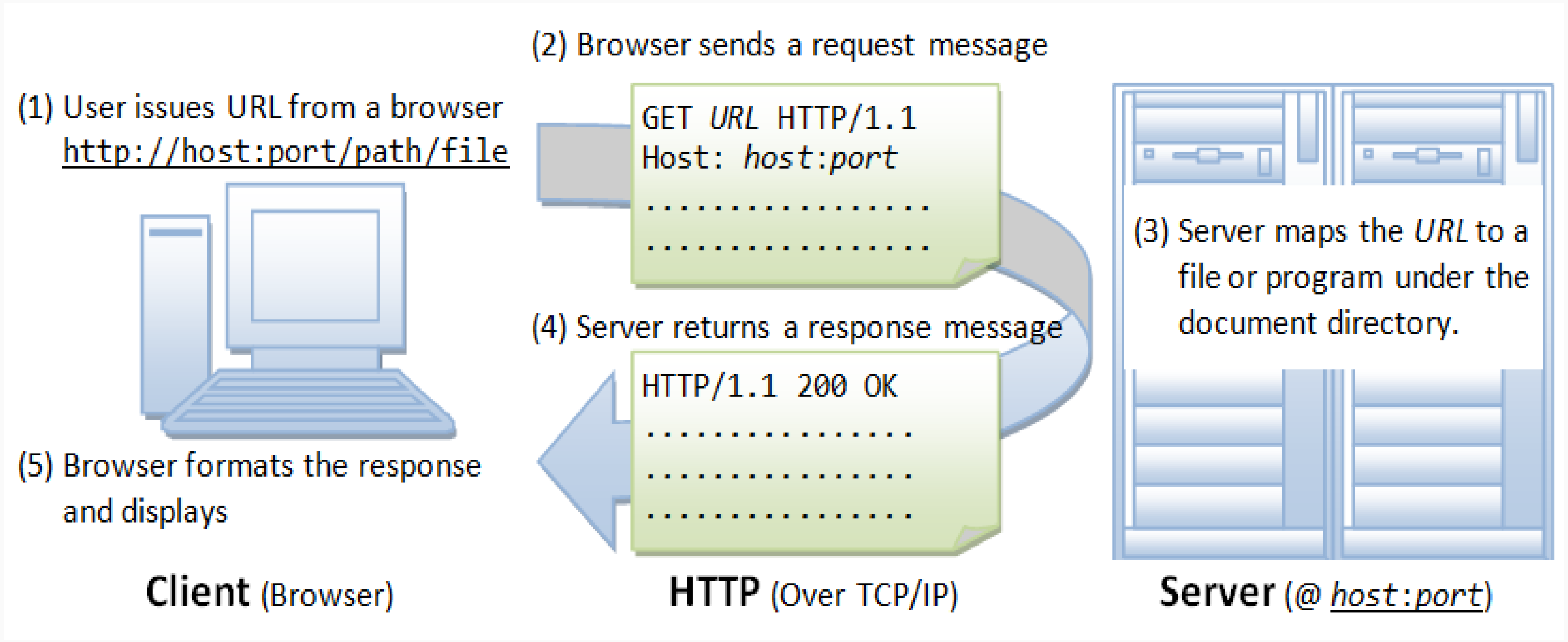
Session

Session is a semi-permanent interactive information interchange, also known as a dialogue, a conversation or a meeting, between two or more communicating devices, or between a computer and user.

Sessions have a unique identifier that maps them to specific users. This identifier can be passed in the URL or saved into a session cookie.

- HTTP cookie
- Session cookie
- Session token

Client Server Interaction



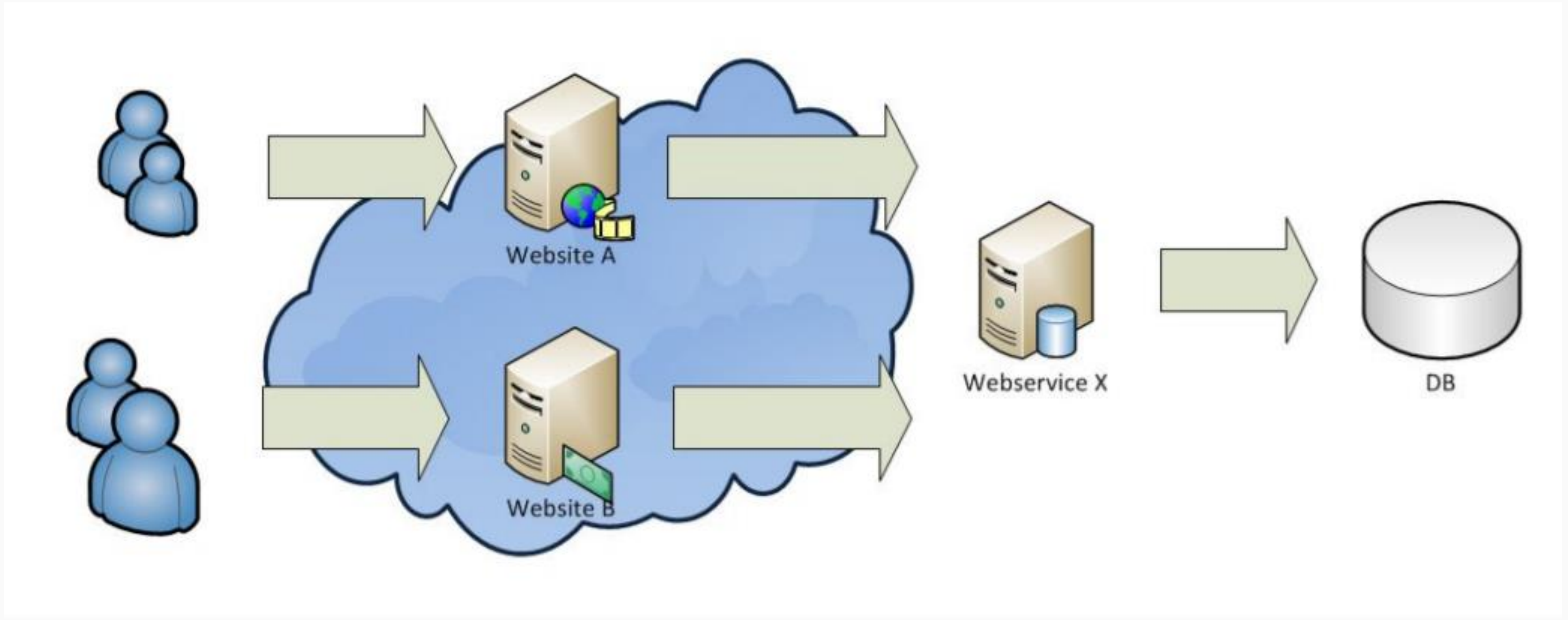
(2) Browser sends a request message
(1) User issues URL from a browser
http://host:port/path/file
GET *URL* HTTP/1.1
Host: *host*:*port*
....
....
(3) Server maps the *URL* to a file or program under the document directory.
(4) Server returns a response message
HTTP/1.1 200 OK
....
....
....
(5) Browser formats the response and displays
Client (Browser)
HTTP (Over TCP/IP)
Server (@ *host*:*port*)

Web Services

Web Services

Definition: Pieces of business logic that can be accessed over the Internet.

You can reuse someone else’s business logic instead of replicating it yourself. This technique is similar to what programmers currently do with libraries of APIs, classes, and components. The main difference is that web services can be located remotely on another server and managed by another company.



Web Services

Benefits of Web Services

- **Web services are simple**: it means they can be easily supported on a wide range of platforms.
- **loosely coupled**: The web service may extend its interface and add new methods without affecting the clients as long as it still provides the old methods and parameters.
- **stateless**: A client makes a request to a web service, the web service returns the result, and the connection is closed. There is no permanent connection. This makes it easy to scale up and out to many clients and use a server farm to serve the web services
- **firewall-friendly**: Firewalls can pose a challenge for distributed object technologies. The only thing that almost always gets through firewalls is HTTP traffic on ports 80 and 443. Because web services use HTTP, they can pass through firewalls without explicit configuration

Disadvantage: there is no support for bidirectional communication, which means the web server cannot call back to a client after the client disconnects.