Networks, HTTP and more

sebinbenjamin

What we'll do learn today

- Recap
- Computer Networks
  - HTTP
  - TCP/IP
    - IP Addresses
    - Sockets, Ports
  - Web and HTTP
- HTTP Request
  - Request Methods
  - Response Message
    - Response Status Codes
- Postman

NPM CLI

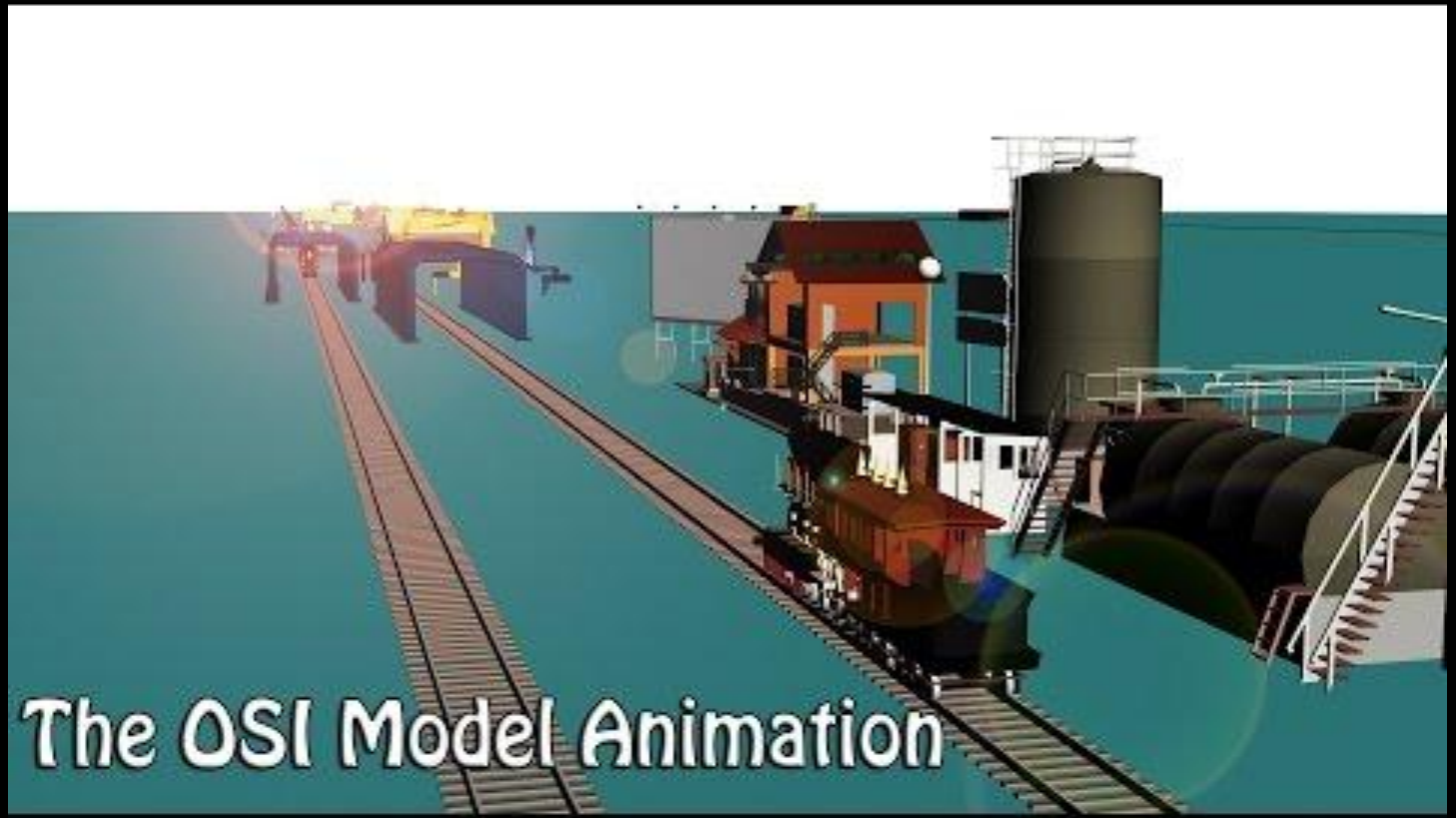
- `npm init` - Create a package.json file
- `npm install` - Install a package
- `npm outdated` - check the registry to see if any (or, specific) installed packages are currently outdated.
- `npm update` - Update an installed package
- `npm uninstall` - Uninstall a package
- `npm ls | list` - List installed packages
- `npm start` - Start a package
- `npm audit` - Run a security audit
- `npm audit fix` - Run a security audit and fix auto-fixable issues

Global Flag `-g`

Refer - NPM CLI Commands

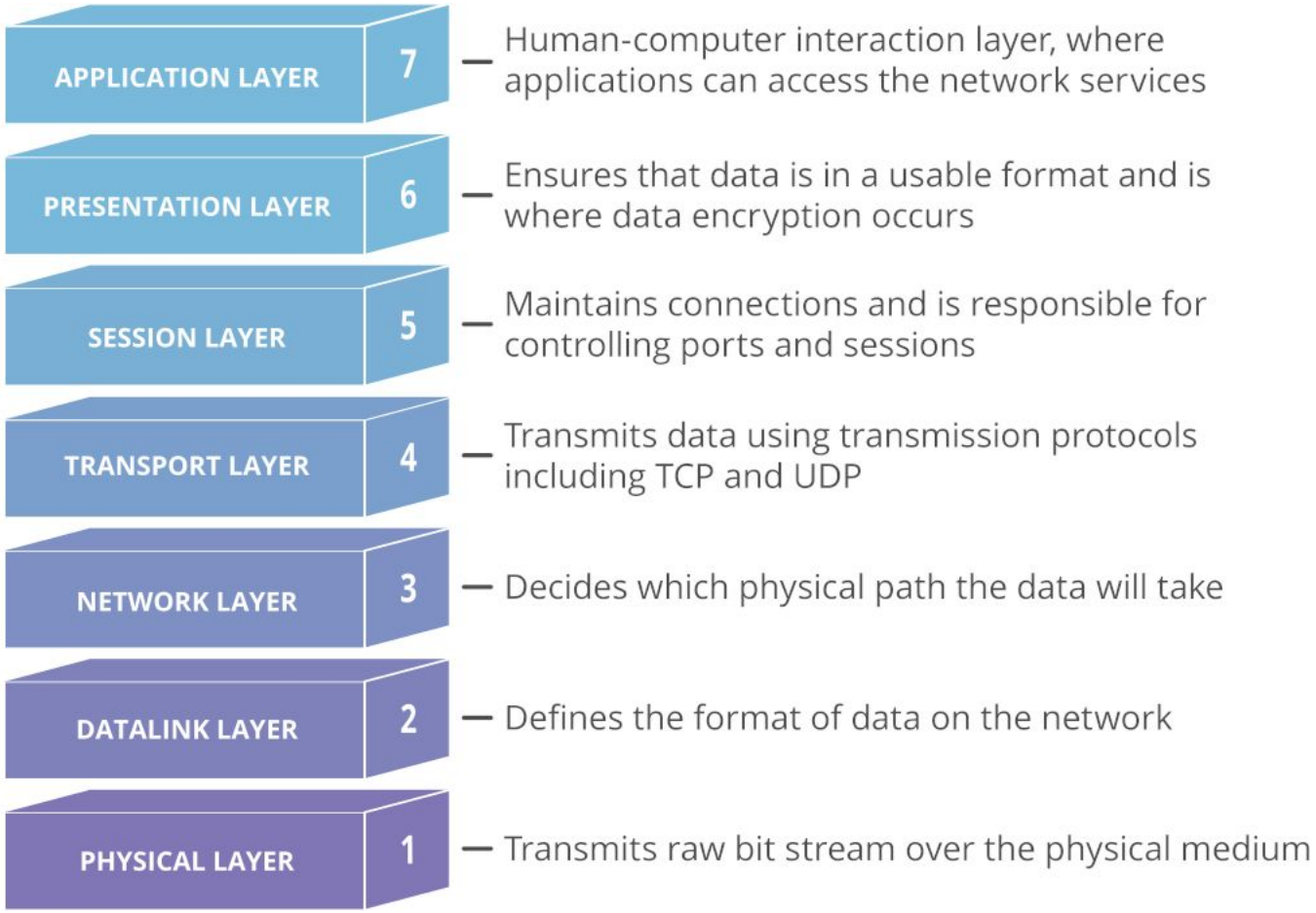
The OSI Layers

- OSI provides a standard for different computer systems to be able to communicate with each other.
- Concept of splitting up a communication system into seven abstract layers, each one stacked upon the last.
- Each layer of the OSI model handles a specific job and communicates with the layers above and below itself.

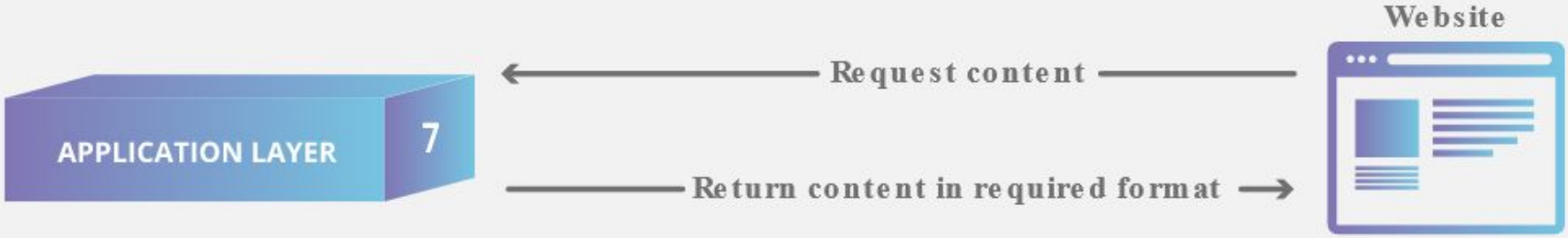


https://youtu.be/-6Uoku-M6oY

https://youtu.be/Ilk7UXzV_Qc?t=54


APPLICATION LAYER
7
Human-computer interaction layer, where applications can access the network services
PRESENTATION LAYER
6
Ensures that data is in a usable format and is where data encryption occurs
SESSION LAYER
5
Maintains connections and is responsible for controlling ports and sessions
TRANSPORT LAYER
4
Transmits data using transmission protocols including TCP and UDP
NETWORK LAYER
3
Decides which physical path the data will take
DATALINK LAYER
2
Defines the format of data on the network
PHYSICAL LAYER
1
Transmits raw bit stream over the physical medium

Application Layer



The Presentation Layer

Encryption

Compression

Translation

The Session Layer

Session of communication

Transport Layer

Segmentation

Transport

Reassembly

The Network Layer

The Data Link Layer

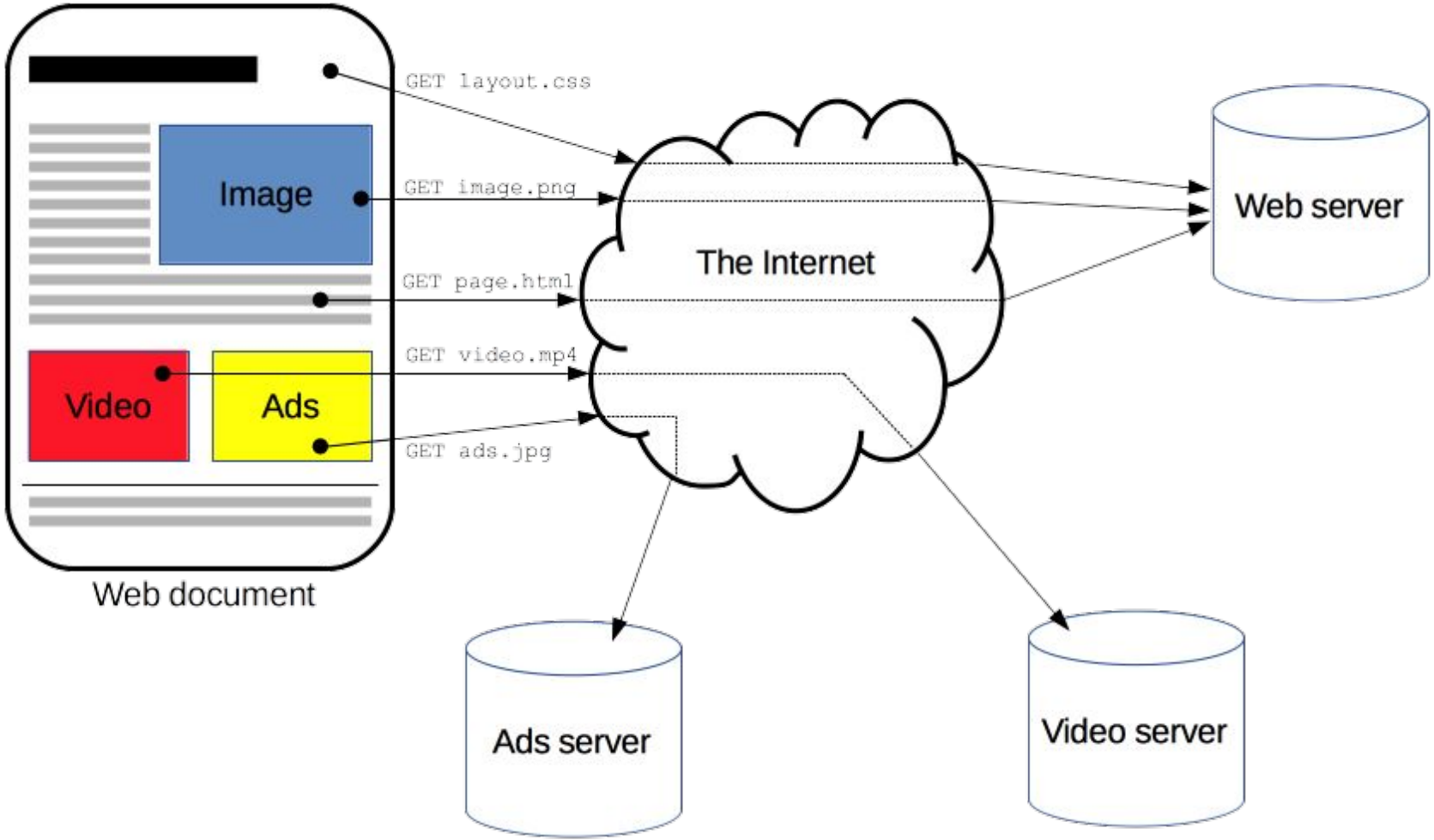
The Physical Layer

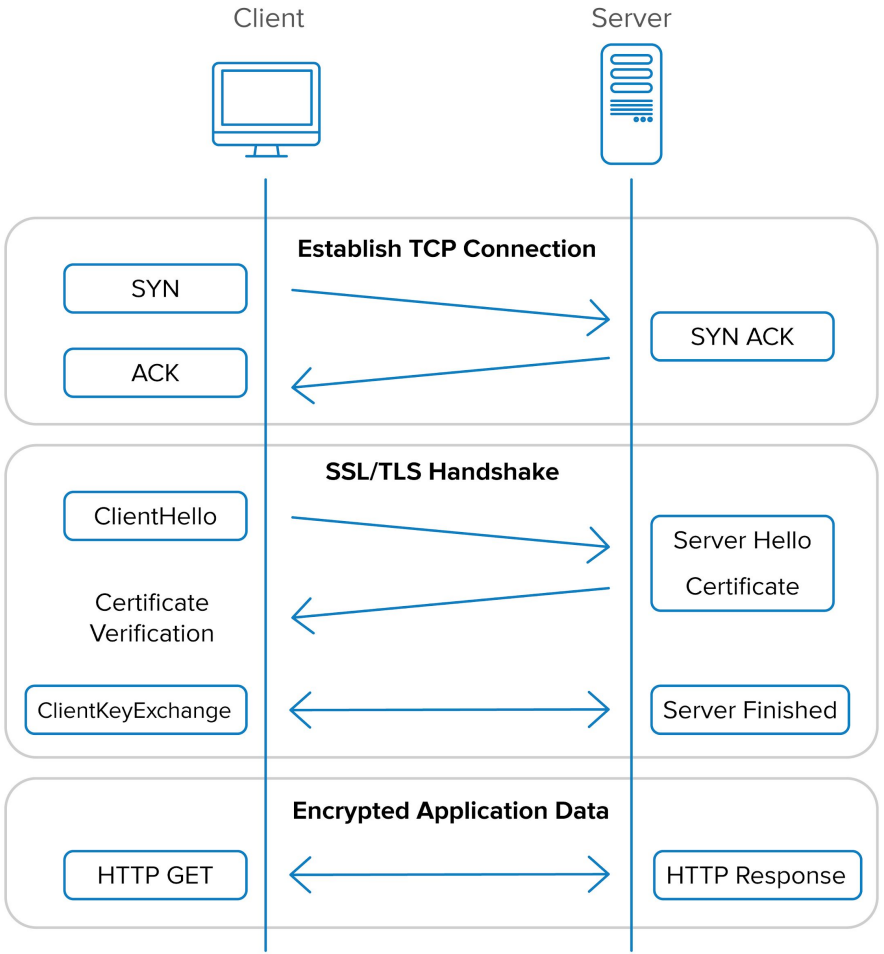
Sending cable | Bitstream | Receiving cable


192.168.1.1
75.75.75.75
What is an IP address?
10.0.1.14
127.0.0.1
200.0.0.1

HTTP

- Hypertext Transfer Protocol is a protocol for **communication** through internet.
- It is the communication standard used by your web browser and the web server
  - It allows fetching of resources - text, images, video’s from the server in an standard way.
- It is a client-server protocol.
  - **Requests** are initiated by the recipient, usually the Web browser
    - The browser is always the entity initiating the request. It is never the server
  - **Responses** given by the server fullfills the requests.


GET layout.css
Image
GET image.png
GET page.html
Video
Ads
GET video.mp4
GET ads.jpg
Web document
The Internet
Web server
Ads server
Video server


Client
Server
Establish TCP Connection
SYN
SYN ACK
ACK
SSL/TLS Handshake
ClientHello
Server Hello
Certificate
Certificate
Verification
ClientKeyExchange
Server Finished
Encrypted Application Data
HTTP GET
HTTP Response

HTTP enables communication between computers by specifying the common rules.

HTTP Messages

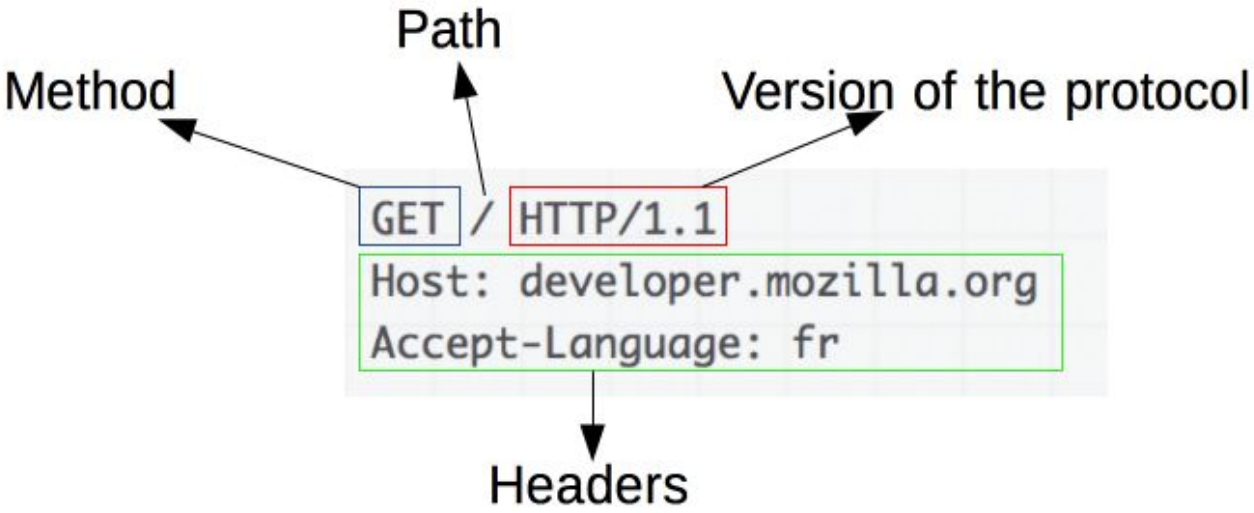
- Messages are used for communication between clients and servers following HTTP
- There are two types of HTTP messages
  - Requests
  - Responses
- https://youtu.be/pHFWGN-upGM

HTTP Request

An HTTP requests consists of the following elements

- An HTTP **method**
- **Path** of the resource to fetch
- **Version** of the HTTP protocol
- Optional **headers**
- **Body**, for some methods


PROTOCOL
PATH
FRAGMENT
http://
www.example.com
/pages/
?id=1&cat=test
#article
DOMAIN
QUERY STRING


Path
Method
Version of the protocol
GET / HTTP/1.1
Host: developer.mozilla.org
Accept-Language: fr
Headers

HTTP Methods

Nouns/Verbs that indicate the desired action to be performed for a given resource.

Commonly used methods

GET method - requests a specified resource/data.

POST method - submit an entity to the specified resource

PUT method - replaces all current with the sent request payload

DELETE method - deletes the specified resource.

https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods

HTTP Status Codes

HTTP response status codes indicate whether a specific HTTP request has been successfully completed. Responses are grouped in five classes:

1. Informational responses (100–199)
2. **Successful responses** (200–299)
3. Redirects (300–399)
4. **Client errors** (400–499)
5. **Server errors** (500–599)

Some common ones

200 OK - success status response code indicates that the request has succeeded

201 Created - The request has succeeded and a new resource has been created.

400 Bad Request - Indicates that the server cannot or will not process the request due to something that is perceived to be a **client error**.

401 Unauthorized - indicates that the request has not been applied because it **lacks valid** authentication **credentials** for the target resource.

404 Not Found - indicates that the server can't find the requested resource.

Some common ones

405 Method Not Allowed - indicates that the request method is known by the server but is not supported by the target resource.

500 Internal Server Error - indicates that the server encountered an unexpected condition that prevented it from fulfilling the request.

HTTP Response

Responses consist of the following elements:

- The ***version*** of the HTTP protocol they follow.
- A ***status code,*** indicating if the request was successful, or not, and why.
- A ***status message***, a short description of the status code.
- HTTP headers, like those for requests.
- Optional - A **body** containing the fetched resource.

HTTP Headers

Headers let the client and the server pass additional information with an HTTP request or response.

Headers can be grouped according to their contexts:

- **General headers -** Apply to both requests and responses, but with no relation to the data transmitted in the body.
- **Request headers** - Contain more information about the resource to be fetched, or about the client requesting the resource.
- **Response headers -** Hold additional information about the response, like its location or about the server providing it.
- **Entity headers** - Contain information about the body of the resource, like its content length or MIME type.

Some common headers

Headers contain meta-information, mainly used for coordination between the client (browser) and the server.

Accept - Content types the client can process; if the field is empty, these are all content types. Eg - text/html

User-Agent - User-Agent of the client (ie, the browser version)

Cache-Control - Whether and how long the object may be kept in the cache.

Allow - Permitted request types for a specific resource.

Date - Time of the response

Some terms

- API : Application Programming Interface
- DNS : Domain Name Server
- WWW : World Wide Web
- HTTP : Hypertext Transfer Protocol
- IP : Internet Protocol
- LAN : Local Area Network | WAN: Wide-Area Network
- URI : Uniform Resource Identifier |URL: Uniform Resource Locator | URN: Uniform Resource Name

Hello World ++

```
const http = require('http');


// Defining the behaviour of the server
const server = http.createServer((req, res) => {
  console.log('Got a request !');
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/html');
  res.setHeader('Content-Language', 'en-US'); // try de-CA
  // sent the Hello world response
  res.end('<h1 style="color:red">Hello, World!</h1>');
});
```

```
const port = 3000;

// Starts the server to listen for requests at a specified port
server.listen(port, () => {
  console.log(`Server running at port ${port}`);
});
```

- Helps you to call API’s easily instead of from a browser or from your frontend code
- Useful for quickly sorting out API issues
- Collaboration and testing
- Lot more …

https://www.postman.com/downloads/

Thank you