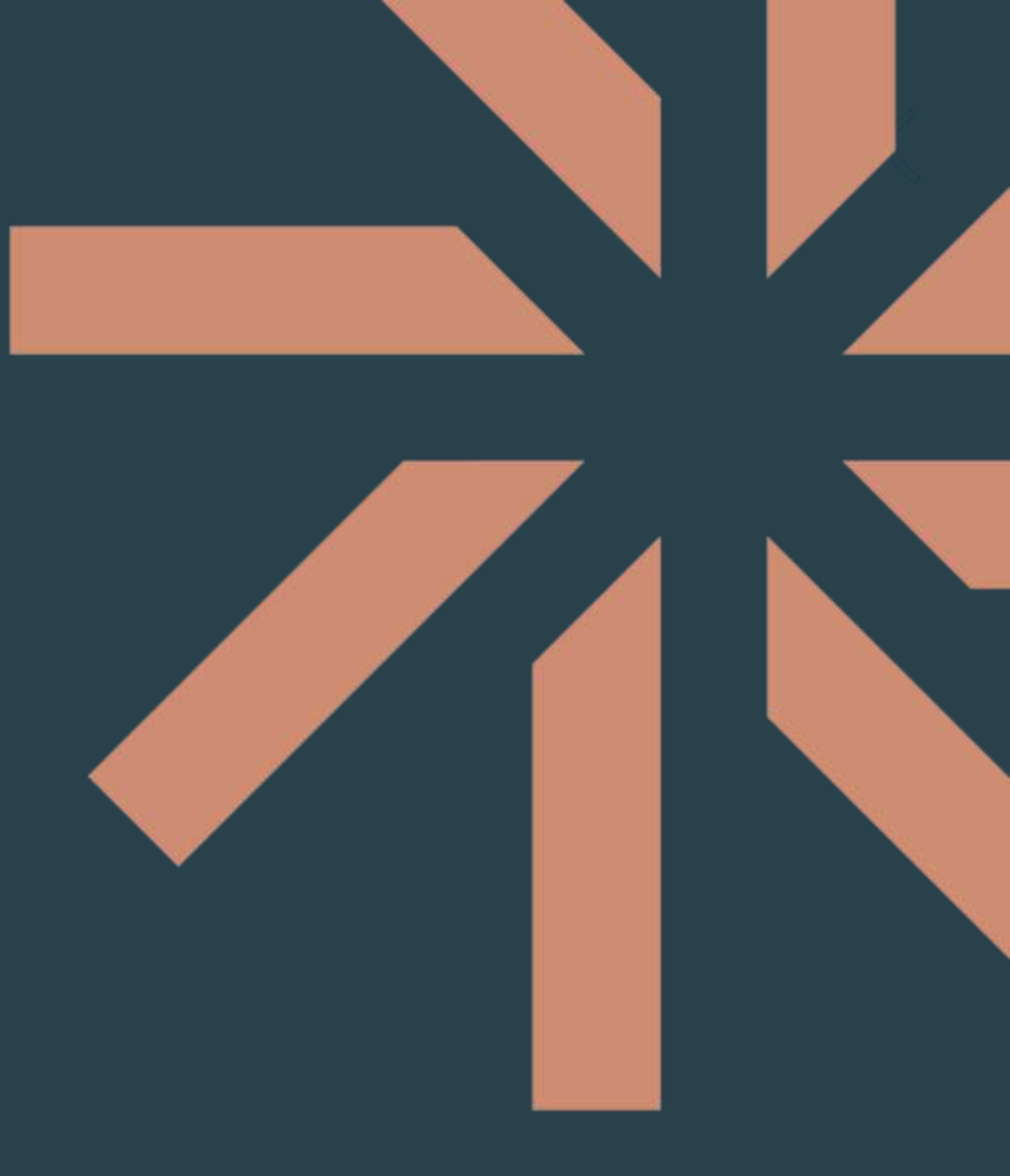
HTTP

Qinshift
Academy

WHAT IS HTTP?

HTTP is a protocol for fetching resources such as HTML documents. It is the foundation of any data exchange on the Web and it is a client-server protocol.

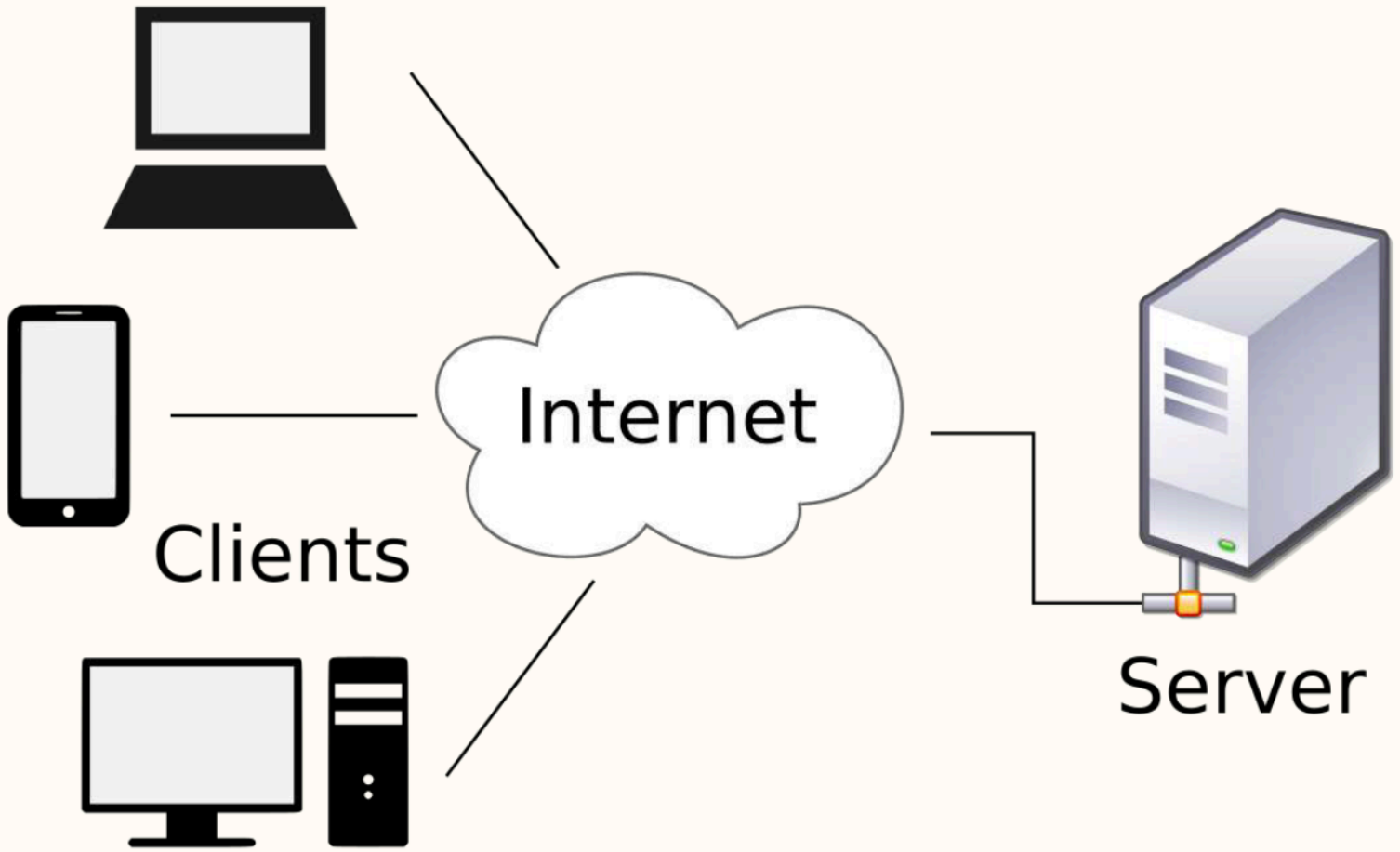
Clients and servers communicate by exchanging individual messages. The messages sent by the client, usually a Web browser, are called requests and the messages sent by the server as an answer are called responses.



COMPONENTS OF HTTP

- HTTP is a client-server protocol: requests are sent by one entity, the user-agent (or a proxy on behalf of it). Most of the time the user-agent is a Web browser, but it can be anything, for example, a robot that crawls the Web to populate and maintain a search engine index.
- Each individual request is sent to a server, which handles it and provides an answer called the response. Between the client and the server there are numerous entities, collectively called proxies, which perform different operations and act as gateways or caches, for example.

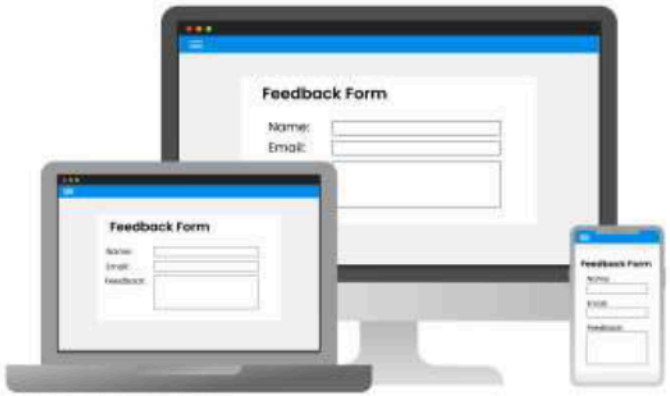
Client & Server



Client & Server

Client

Client-side code runs on the user's browser and is untrusted.

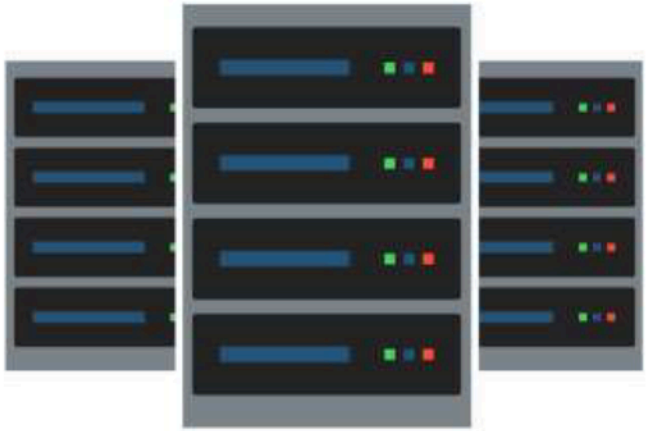


Responsible for:

- ✔ Displaying the UI
- ✔ Interactivity

Server

Server-side code runs on a server and is unseen by users.



Responsible for:

- ✔ Controlling central resources
- ✔ Permissions checks
- ✔ Handling secret information

Client: the user-agent

The user-agent is any tool that acts on behalf of the user. This role is primarily performed by the Web browser, but it may also be performed by programs used by engineers and Web developers to debug their applications.

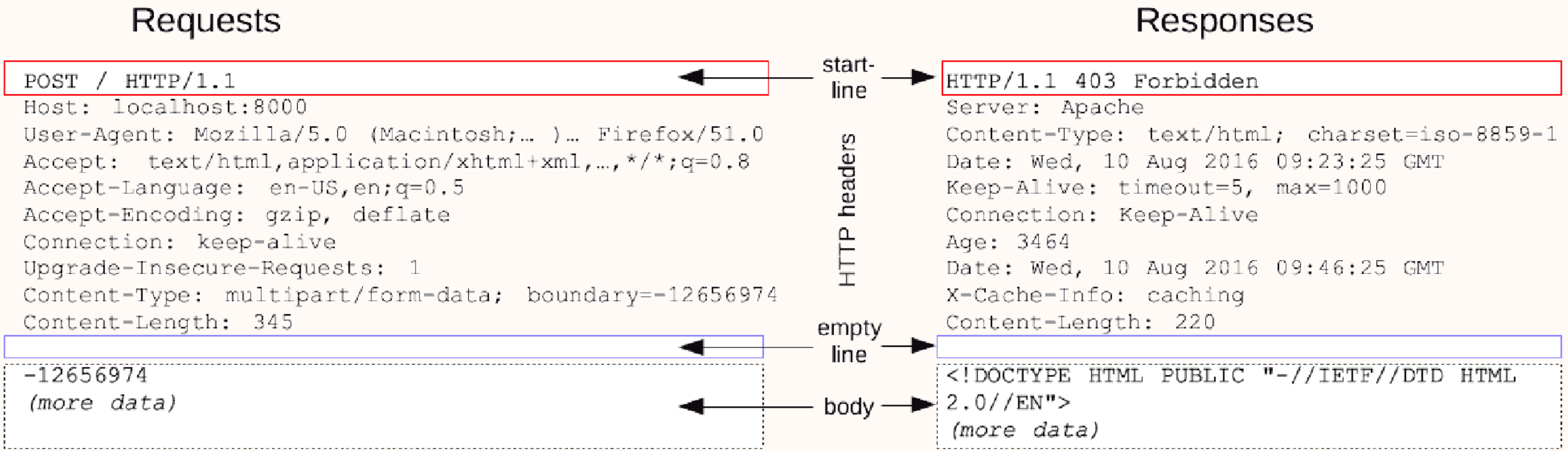
To display a Web page, the browser sends an original request to fetch the HTML document that represents the page.

The web server

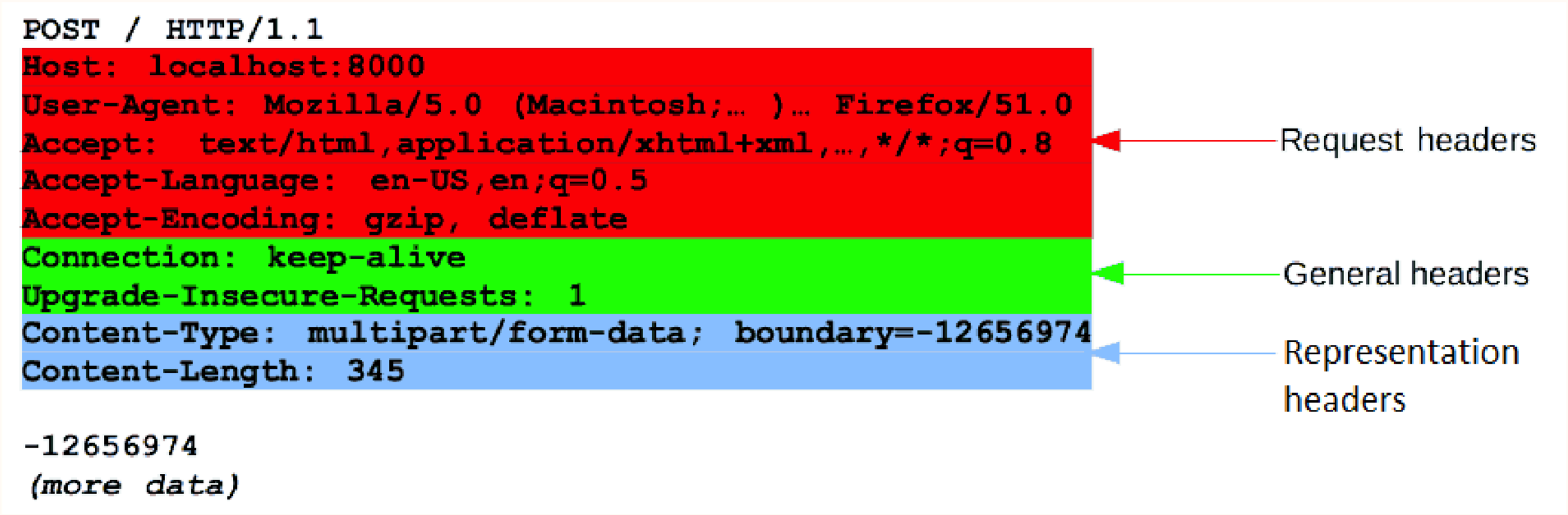
On the opposite side of the communication channel is the server, which serves the document as requested by the client. A server appears as only a single machine virtually; but it may actually be a collection of servers sharing the load (load balancing), or a complex piece of software interrogating other computers (like cache, a DB server, or e-commerce servers), totally or partially generating the document on demand.

HTTP Messages

HTTP messages, as defined in HTTP/1.1 and earlier, are human-readable. There are two types of HTTP messages, requests and responses, each with its own format.



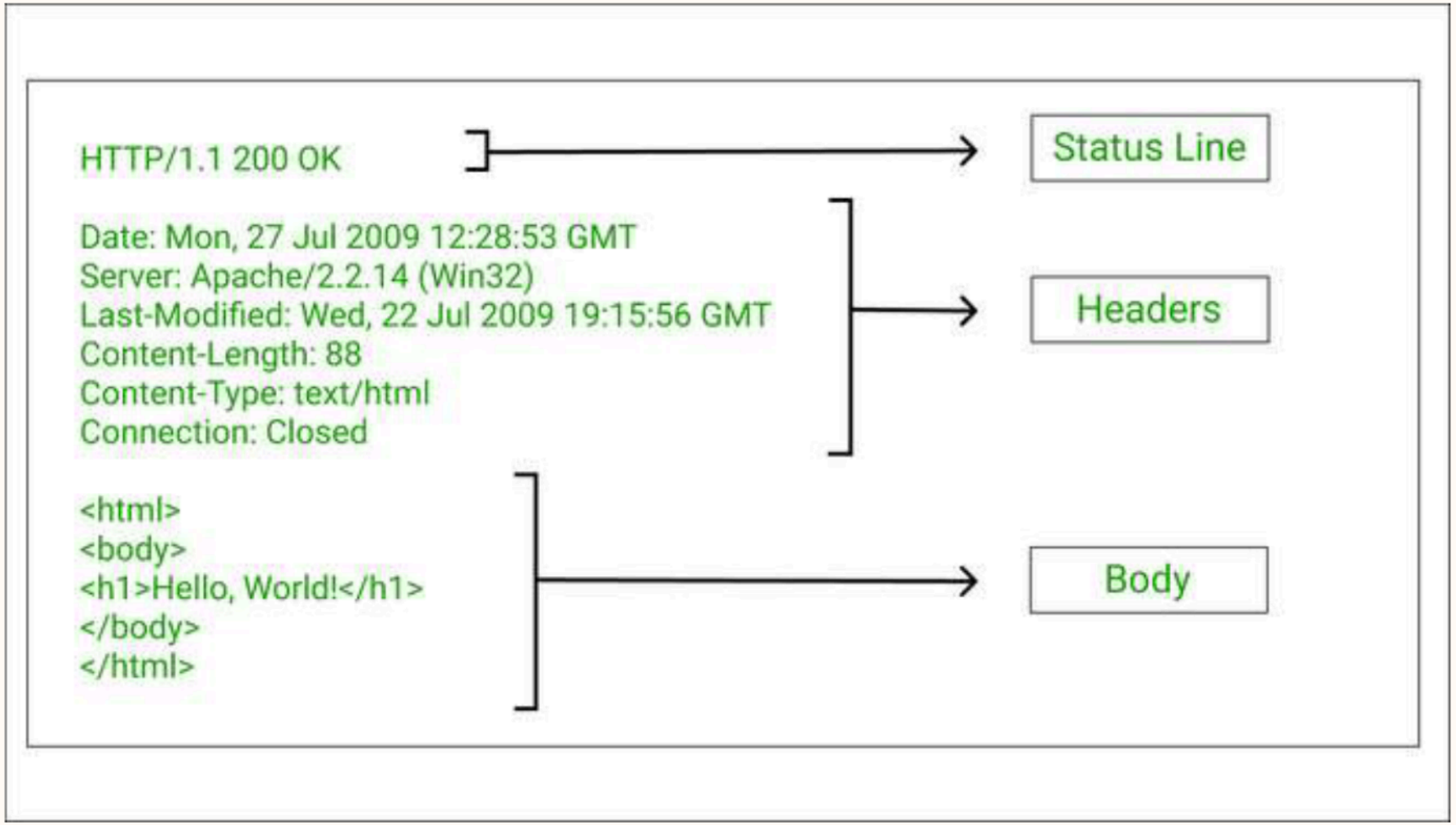
HTTP Messages



Requests consist of the following elements

- An HTTP method, usually a verb like GET, POST, or a noun like OPTIONS or HEAD that defines the operation the client wants to perform. Typically, a client wants to fetch a resource (using GET) or post the value of an HTML form (using POST), though more operations may be needed in other cases.
- The path of the resource to fetch; the URL of the resource stripped from elements that are obvious from the context, for example without the protocol (http://), the domain (here, developer.mozilla.org), or the TCP port (here, 80)
- The version of the HTTP protocol.
- Optional headers that convey additional information for the servers.
- A body, for some methods like POST, similar to those in responses, which contain the resource sent.

HTTP Responses



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 non-authoritative short description of the status code.
- HTTP headers, like those for requests.
- Optionally, a body containing the fetched resource.

HTTP Status Codes

HTTP response status codes indicate whether a specific HTTP request has been successfully completed.

Responses are grouped in five classes:

- Informational responses (100 – 199)
- Successful responses (200 – 299)
- Redirection messages (300 – 399)
- Client error responses (400 – 499)
- Server error responses (500 – 599)

HTTP Status Codes

2xx Success

200	Success / OK

3xx Redirection

301	Permanent Redirect
302	Temporary Redirect
304	Not Modified

4xx Client Error

401	Unauthorized Error
403	Forbidden
404	Not Found
405	Method Not Allowed

5xx Server Error

501	Not Implemented
502	Bad Gateway
503	Service Unavailable

HTTP status ranges in a nutshell:

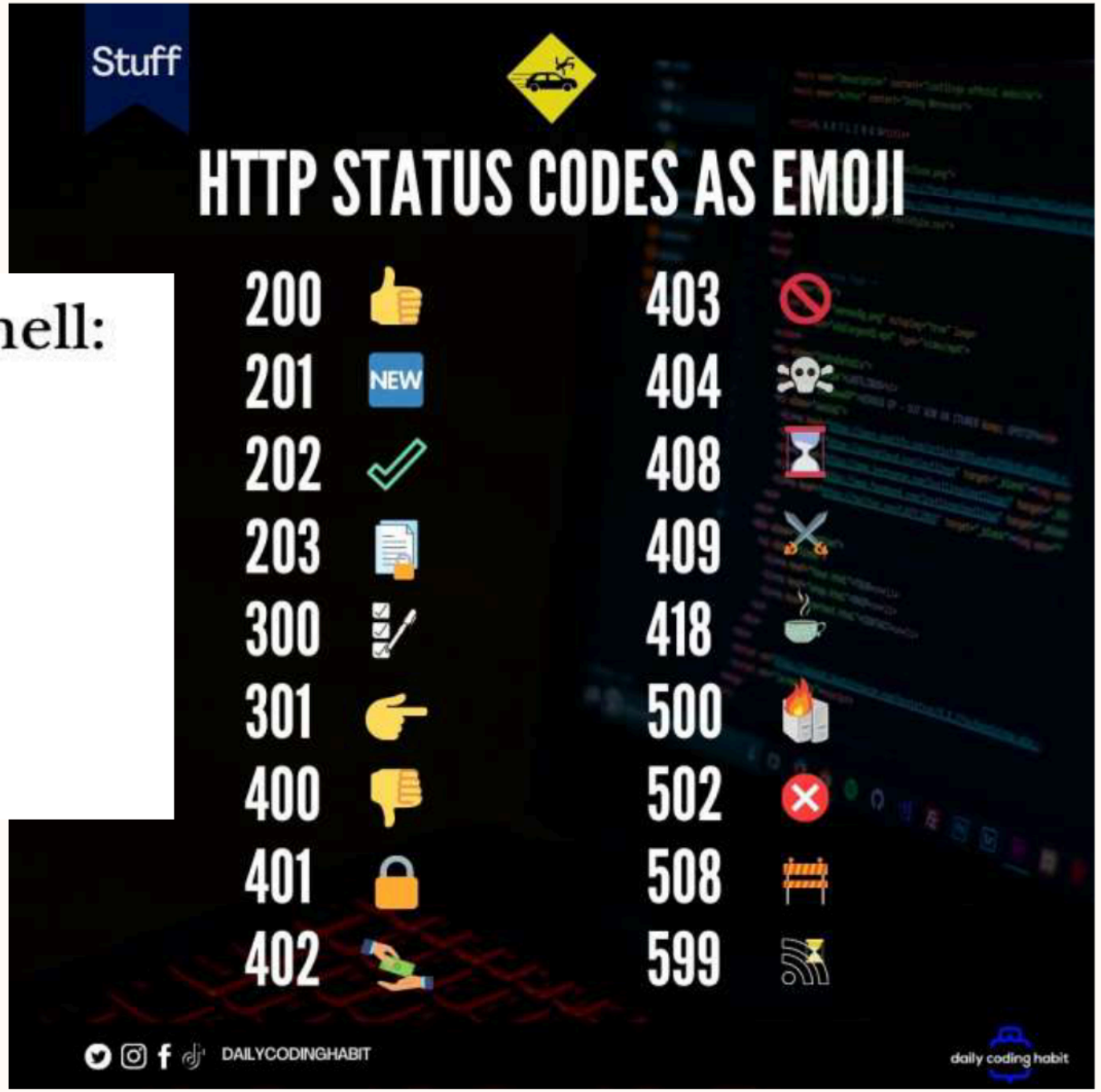
1xx: hold on

2xx: here you go

3xx: go away

4xx: you messed up

5xx: I messed up



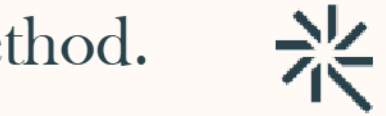
HTTP MODULE IN NODE

To make HTTP requests in Node.js, there is a built-in module HTTP in Node.js to transfer data over the HTTP. To use the HTTP server in node, we need to require the HTTP module. The HTTP module creates an HTTP server that listens to server ports and gives a response back to the client.

```
import http = require('http') //Common JS
```

```
import * as http from 'http'; //ES 6
```

We can create a HTTP server with the help of http.createServer() method.



```
// Create a server
http.createServer((request, response)=>{

    // Sends a chunk of the response body
    response.write('Hello World!');

    // Signals the server that all of
    // the response headers and body
    // have been sent
  response.end();
})
.listen(3000); // Server listening on port 3000
```

QUESTIONS?

You can find us at:

anetastankovskaane@gmail.com

igorveic7@gmail.com

Qinshift Academy