HTTP Concepts: Conceptualized

Objectives

- Explain what HTTP is & why it's useful.
- Explain what an HTTP request & response is.
- Diagram the HTTP request-response cycle.
- Send HTTP requests and receive HTTP responses for HTML.
- Explain what JSON is & why it's useful.
- Send HTTP requests and receive HTTP responses for JSON.

HTTP: HyperText Transfer Protocol

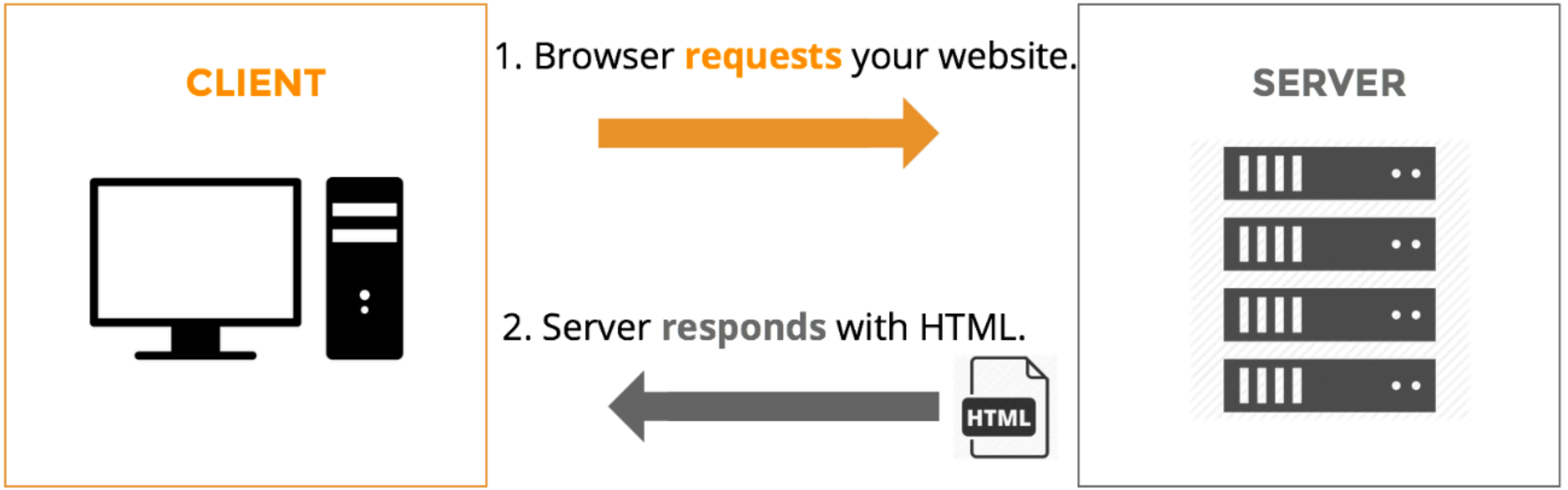
Hypertext: Text has uses "hyperlinks" to other documents
HTML: HyperText Markup Language
WWW: All the public hypertexts in the world
HTTP: A protocol used for transferring HTML and other data.

A **protocol** is a set of rules that allow two networked devices to transmit data.

Sir Tim Berners-Lee and his team at CERN created HTTP/HTML in '89/'90 as well as the first browser Mosaic, thereby starting the WWW.

HTTP

- HTTP is based on a request–response scheme.
- It shuttles messages between two programs called a client and a server. In the case of the WWW, the browser is the client and the web server (or server app) is the server.
- HTTP is useful because it provides a uniform interface for communication even as clients and servers are substituted.


CLIENT
1. Browser requests your website.
2. Server responds with HTML.
HTML
SERVER

HTTP Request

When a Client sends a message to a Server on behalf of the user, this is called a request.

Examples of Clients:
- Browsers
- Search engine web crawlers
- Desktop apps (cURL, wget, others)
- Mobile apps

HTTP Request Anatomy

1. A method (aka verb, e.g. GET, POST, PUT, DELETE)
2. A path
3. An HTTP version
4. Key-Value pairs called headers
5. An optional body

HTTP Request Example

```
GET / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: some-app.herokuapp.com
User-Agent: HTTPie/0.9.3
```

T&T: Can you identify the parts of the above HTTP request?

Request Methods

GET: Retrieve a resource, like HTML or CSS files from a server
POST: Send data, like form field values/data, to a server
PUT: Update a resource on a server
DELETE: Delete a resource on a server

QUESTION: When does a web browser make GET requests? When does it make POST requests?

HTTP Response

After the web server receives a request and does some processing, it replies with an HTTP response.

Popular web servers that generate such responses include Apache, Nginx, Node.js, and Python's SimpleHTTPServer.

HTTP Response Anatomy

1. An HTTP version
2. A status code
3. Key-Value headers
4. An optional body

HTTP Response Example

```
HTTP/1.1 200 OK
Access-Control-Allow-Origin: *
Connection: keep-alive
Content-Length: 631
Content-Type: text/html; charset=utf-8
Date: Sun, 22 May 2016 22:54:23 GMT
Etag: W/"277-ENWB837FwX/qicQv2cu/qA"
Server: Cowboy
Vary: Accept
Via: 1.1 vegur

<!DOCTYPE html>
<html>
  <head><title>Student Roster</title></head>
  <body><content></content></body>
</html>
```

T&T: Looking at the last slide, can you identify the parts of an HTTP response?

HTTP Response Status Codes

Each status code informs the client how the request was handled.[1]

Status Code Group	Description
1XX	Accepted, ready for the next one.
2XX	Accepted, the server's work is complete.
3XX	Accepted, but additional client work is needed.
4XX	Accepted, but there was an error on the client.
5XX	Accepted, but there was an error on the server.

[1] www.httpstatuses.com

T&T: The most common status codes are 200, 302, 304, 404, and 500. What do they mean?

Break

Request-Response Cycle

Wesley Reid Sequence Diagram

HTML Request/Response

cURL, wget, and HTTPie are all command line HTTP clients. Just like your browser they make requests and display responses from a server, except they don't do any page rendering-- they output the raw response.

Let's Try: cURL, HTTPie

```
curl -v -X GET https://fs-student-roster.herokuapp.com/students
```

```
brew install httpie
http -v GET https://fs-student-roster.herokuapp.com/students
```

JSON What

JavaScript Object Notation (JSON)
- A common data exchange format on the Web
- Is a string that looks very similar to native objects and arrays
- JSON strings must be parsed in order for it to be useable as objects in JS
- We call this a "serialized" format

NOTE: Both JSON strings and JSON object keys must be wrapped in double quotes ""

```
{
  "avatar": "https://i.imgur.com/KlycRG5.jpg",
  "hobby": "Motherhood",
  "name": "Daenerys Targaryen"
}
```

JSON Array of Objects

```
[
  {
    "avatar": "https://i.imgur.com/KlycRG5.jpg",
    "hobby": "Motherhood",
    "name": "Daenerys Targaryen"
  },
  {
    "avatar": "https://i.imgur.com/fFMusdC.png",
    "hobby": "Traveling",
    "name": "Tyrion Lannister"
  }
]
```

JSON Why

Why bother converting our objects/arrays to strings (serialization)?

- More easily stored in a file, in a database, or transmitted across a network
- A receiving program can parse this string and the object/array is perfectly reconstructed
- Allows the programs involved in the sending/receiving to be very different, even written in different languages
- The data remains intact from end to end

JSON Request/Response

Let's Try:

```
http -vj GET http://fs-student-roster.herokuapp.com/students
```

T&T: What differences can you spot in the request/response compared to the previous HTML example?

Related Network Concepts

IP Address: Required for communicating on the Internet
- Private: 192.168.x.x, 10.x.x.x
- Public: All others

DHCP: Service which dynamically assigns IP addresses
DNS: Domain Name Servers: Internet equivalent of a phone book
localhost/127.0.0.1: A name and IP that refers to this machine

Ports:

- A network service (aka daemon, server) is open for communication on a particular port(s). E.g. 80=HTTP, 21=FTP
- There are 65,535 possible ports to use
- Root access is required for ports <1024

SSL/TLS: Secure Sockets Layer & Transport Layer Security

- Allows for encrypted HTTP traffic. Banks, and any login/auth page
- You are using this when you see HTTPS:// in browser URL, and lock icon