Construction of User Interfaces (SE/ComS 319)

Jinu Susan Kabala

Department of Computer Science

Iowa State University, Fall 2021

BASICS-NETWORKING

Outline

- Server and client
- Web server and clients
- HTTP Protocol

Terms and definitions

SERVER AND CLIENT

Server

- Program that **provides SERVICES** (i.e. useful functionality). Typically, keeps running forever.
  - Typically, *the computer* that runs the server program is called a server. Many server programs can be running on a single computer.
- Examples of services:
  - database services
  - runs a web site (web servers)

Client

- Program that connects to a SERVER computer - and then to a program that provides services and **USES those services**.
  - Typically, the computer that runs the client program is called a client!
- Multiple clients can typically connect to a server
- Examples:
  - Web browsers on a computer connect to web servers on other computers and is provided with web pages.

MAC ID – IP Address – Hostname

- **MAC ID** is a unique id that is HARD-CODED on every computer (or internet capable device).
  - Already there when you buy the device.
  - Example: c8:bc:c8:9b:c4:0f for ethernet card of a computer.
  - Used by lower protocols to uniquely identify a device.
- **IP ADDRESS** is an address assigned to **computers connected to the internet.**
  - Typically assigned when connecting to the internet.
  - Example: 129.186.252.23
- Unlike IP address, **HOSTNAME** is a human-readable address (like www.google.com). Servers typically have hostnames.

DNS – Localhost

- **DNS (Domain Name Server) –** is like a phone book.
  - Maps Hostnames to IP addresses (many to one)
  - When you want to connect to a website by typing in a hostname, your computer will find the IP address by asking the DNS.

- **Localhost –** each computer can use the hostname *localhost* to refer to itself!

Port

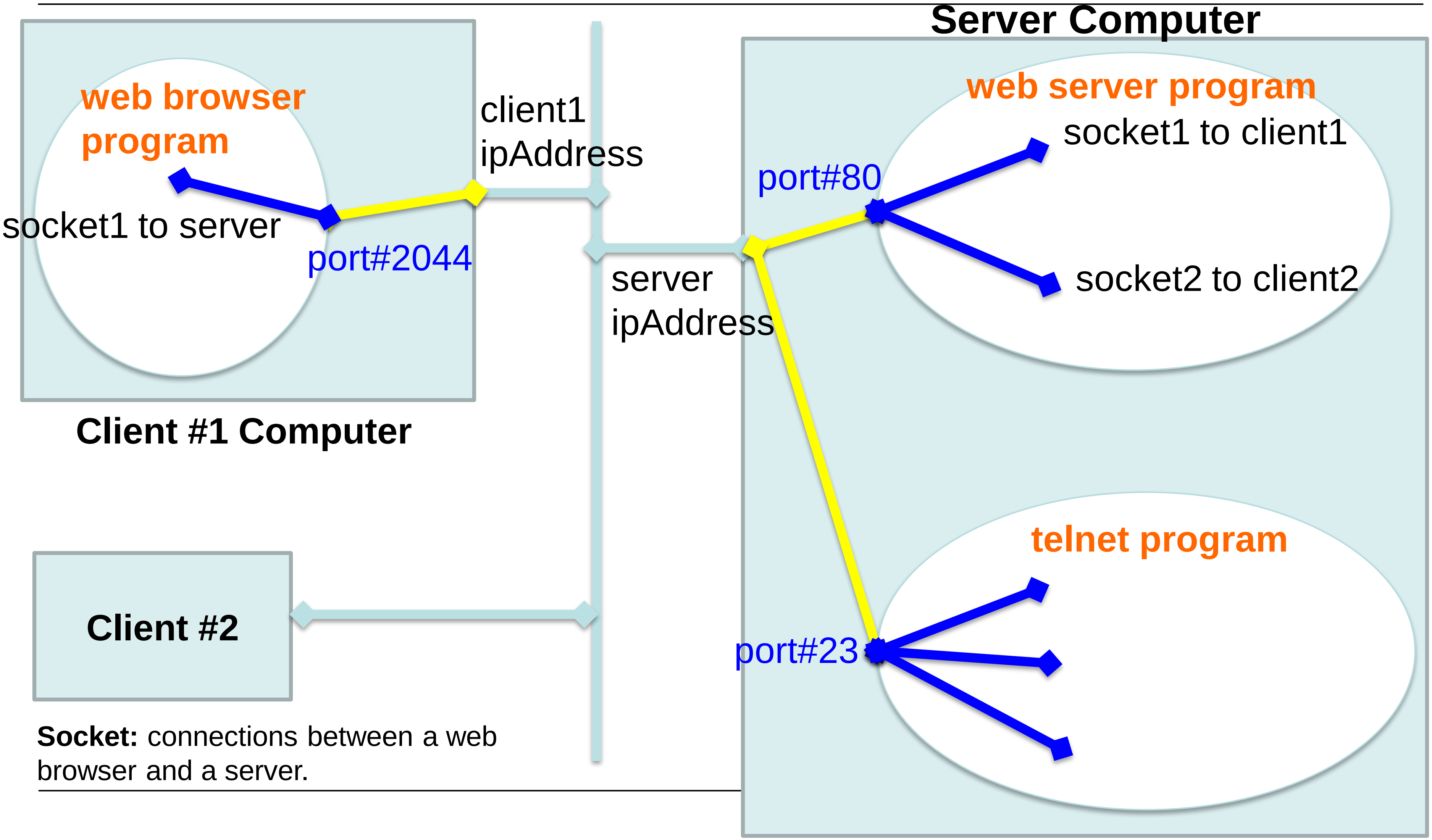
- This is a **NUMBER** that refers to a **specific process** running on a computer.
- Many port numbers are reserved:
  - 80: http
  - 23: telnet (communication for computers on internet/local area networks)
  - 22: ssh
  - 110: pop3 (for email delivery)
- You will be able to create ports only from 1024 onwards.
- Once a port is being **used by a server**, you cannot use that same port for other programs.
- Multiple clients can talk to a server through that port.

Socket

- A socket contains connection information between two computers
- LOCAL ADDRESS
  - local computer's IP address
  - local program's port#
- REMOTE ADDRESS
  - remote computer's IP address
  - remote program's port#
- PROTOCOL
  - means the "LANGUAGE" or "RULES" that the two computers will use to communicate.
  - typically this is TCP/IP protocol.

Socket = host name (IP) and port number

Server and client connection



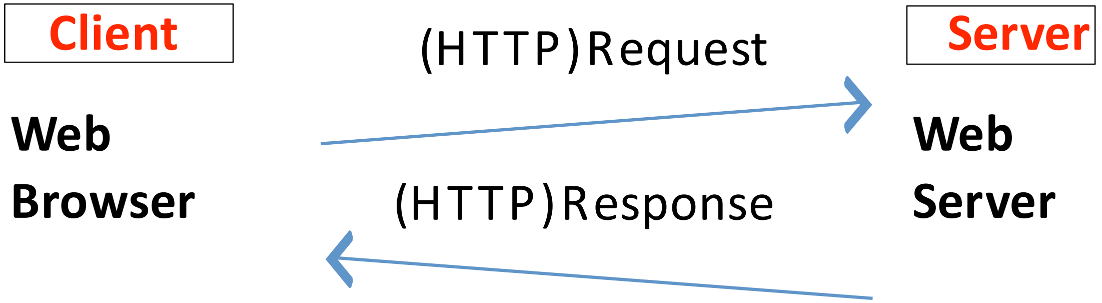
Socket: connections between a web browser and a server.

Demonstration

- Telnet is disabled on most servers for security reasons
- Server @ CS Dep ISU
    - To login to pyrite, you must use the Secure Shell (ssh)
    - ssh <ISU NetID>@pyrite.cs.iastate.edu
    - **Ctrl - d** usually allows you to exit the ssh session normally (logout)
- Condo Cluster @ ISU (High-performance computing cluster)
    - ssh <ISU NetID>@condo2017.its.iastate.edu

WEB SERVER AND CLIENTS

Web server and client



Client

Web Browser

(HTTP)Request

(HTTP)Response

Server

Web Server

HTML
CSS
JavaFX
Plugins
(Java)Script
Applets
…

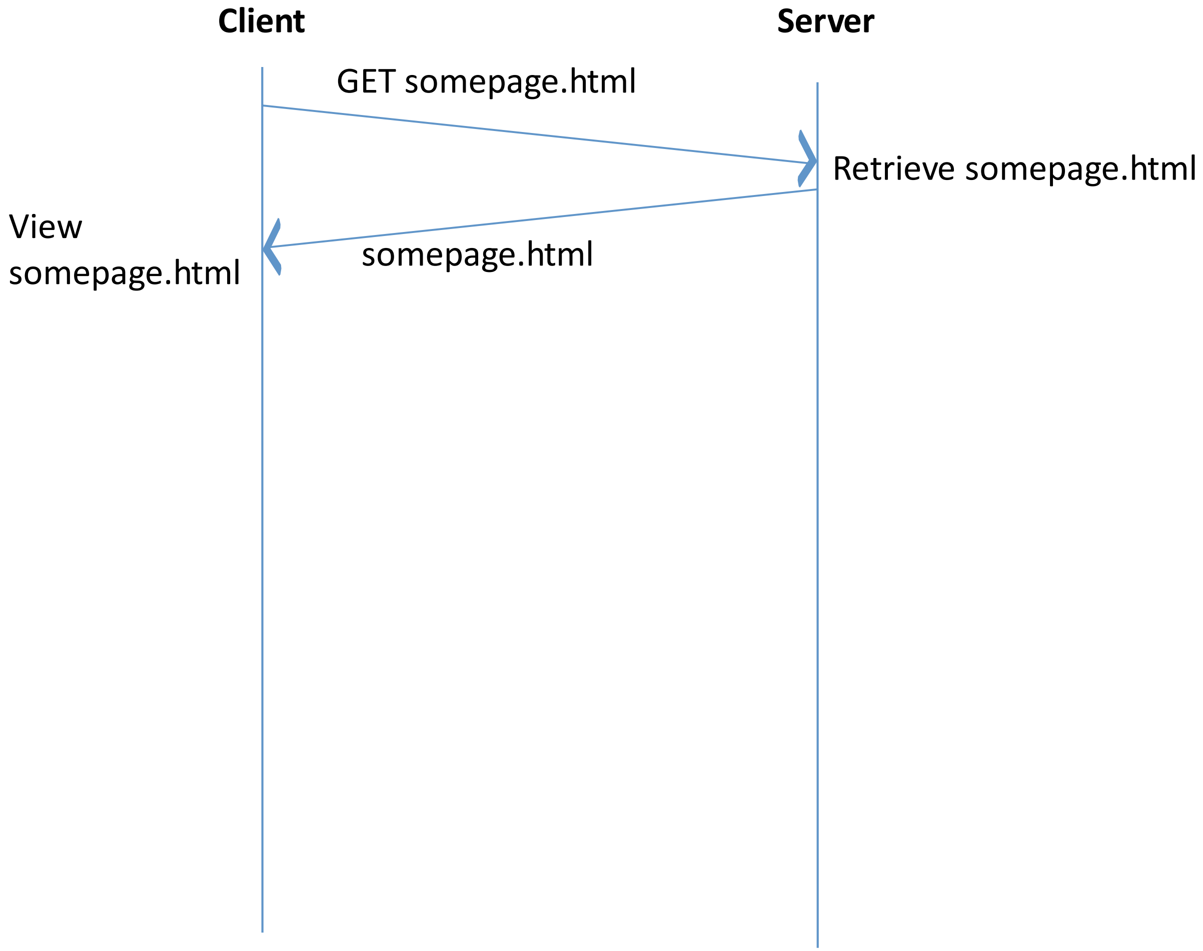
CGI
Node.js
ASP/JSP
PhP
Servlets
…

HTTP (Hypertext Transfer Protocol): HTTP is a client-server application-level protocol. It typically runs over a TCP/IP connection.

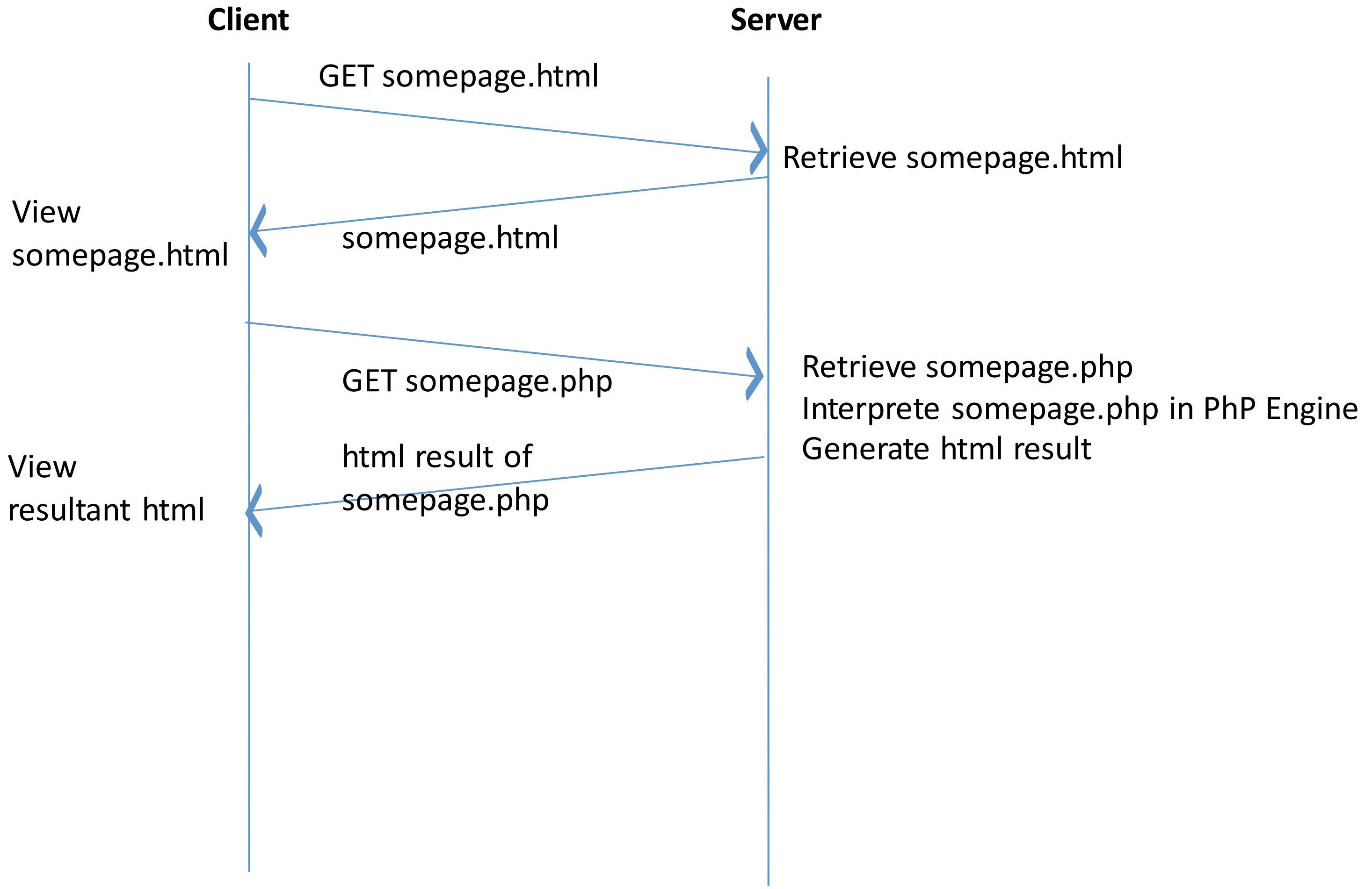
Web server and client – HTTP request

- Web-client and Web-server communicates using HTTP protocol
  - Client can send a HTTP request: method “**`get`**" or “**`post`**"
  - Server can read a **HTTP request** and produce **HTTP response**
- Server-side programs should be capable of reading HTTP request and producing HTTP response
- Command: **GET** *request-URI HTTP-version*
  - e.g. **GET /index.html HTTP/1.0**

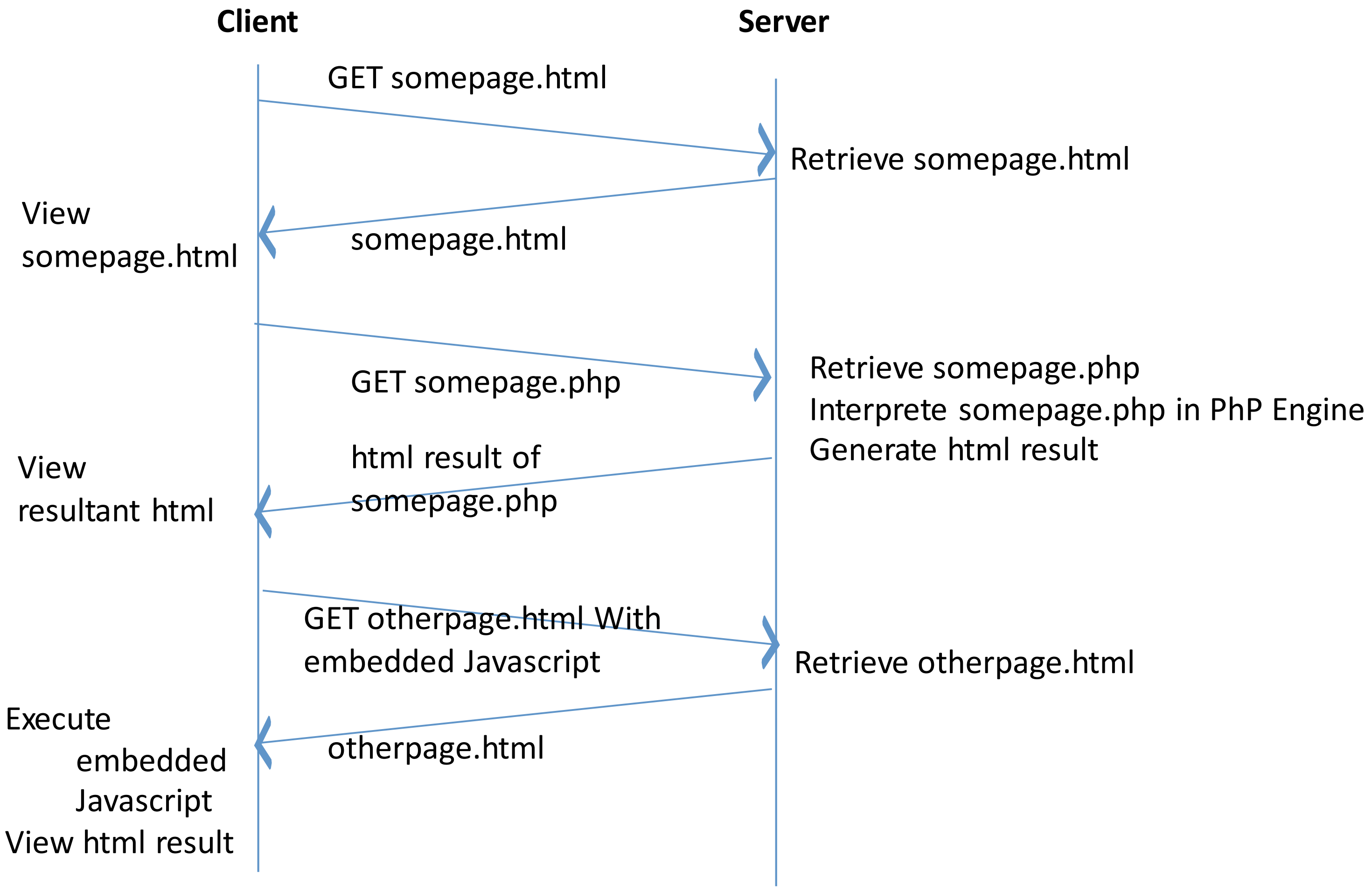
Web server and client interaction (1)



Web server and client interaction (2)



Web server and client interaction (3)



Client

Server

GET somepage.html

Retrieve somepage.html

View
somepage.html

somepage.html

GET somepage.php

Retrieve somepage.php
Interprete somepage.php in PhP Engine
Generate html result

html result of
somepage.php

View
resultant html

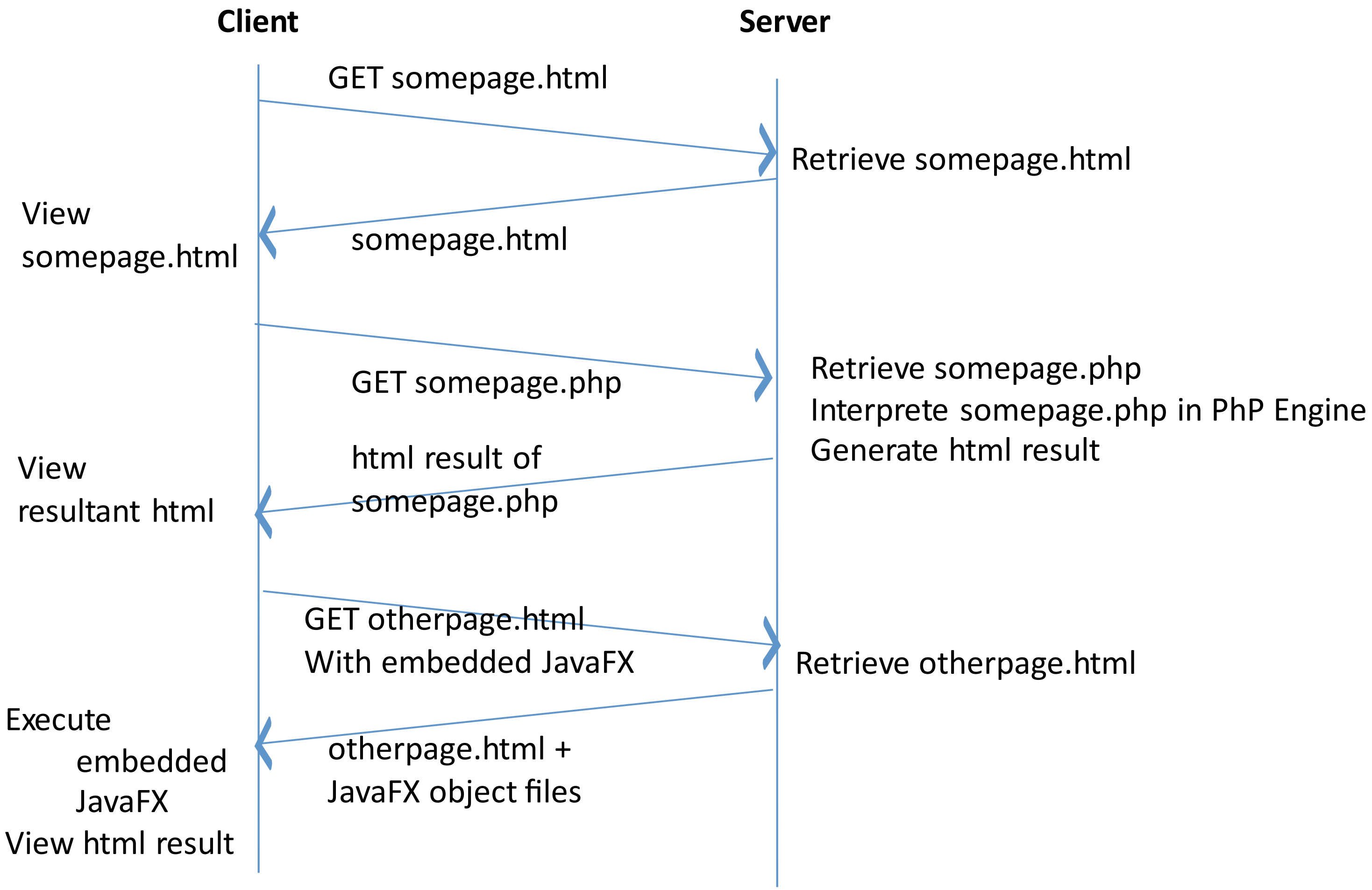
GET otherpage.html With
embedded Javascript

Retrieve otherpage.html

otherpage.html

Execute
embedded
Javascript
View html result

Web server and client interaction (4)



Client

Server

GET somepage.html

Retrieve somepage.html

View
somepage.html

somepage.html

GET somepage.php

Retrieve somepage.php
Interprete somepage.php in PhP Engine
Generate html result

View
resultant html

html result of
somepage.php

GET otherpage.html
With embedded JavaFX

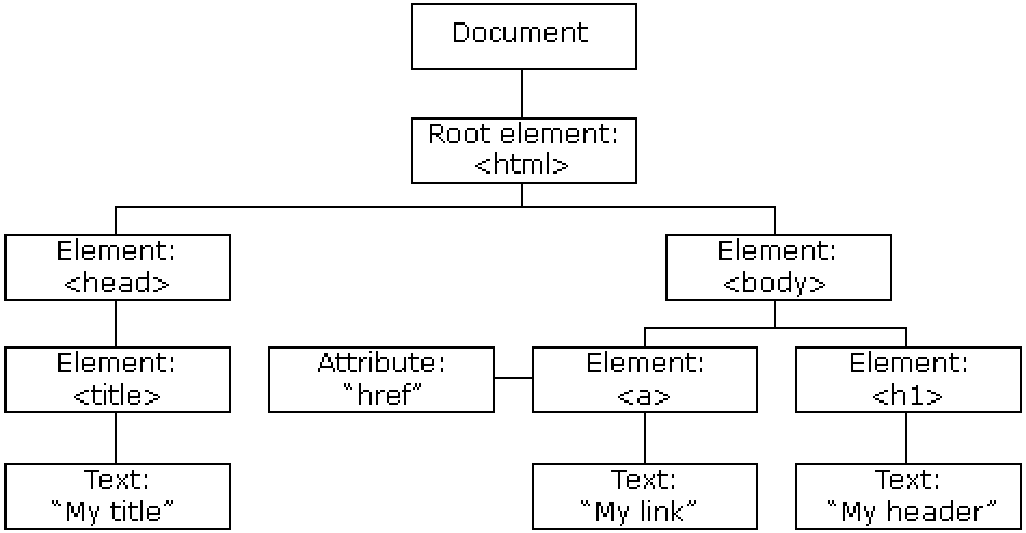
Retrieve otherpage.html

Execute
embedded
JavaFX
View html result

otherpage.html +
JavaFX object files

Demonstration

- HTML
- DOM (Document Object Model) : An application programming interface (API) for HTML
  - It defines the logical structure of documents and the way a document is accessed and manipulated
  - Demo:
  - https://www.w3schools.com/



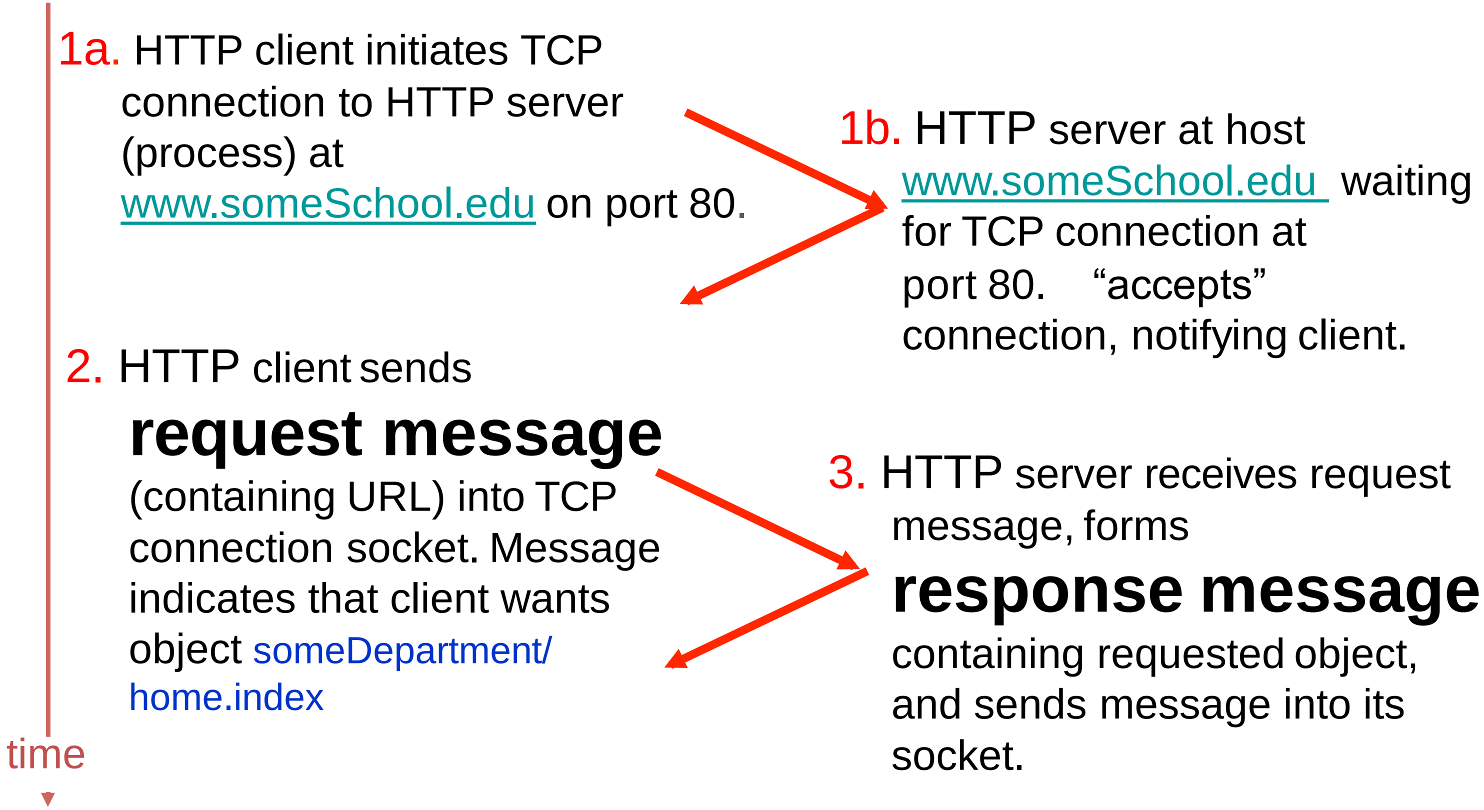
Source: https://www.w3.org

- HTML, DOM, etc. in Safari/Chrome

HTTP PROTOCOL

Normal HTTP requests (1)

Suppose user enters URL:
www.someSchool.edu/someDepartment/home.index



1a. HTTP client initiates TCP connection to HTTP server (process) at www.someSchool.edu on port 80.

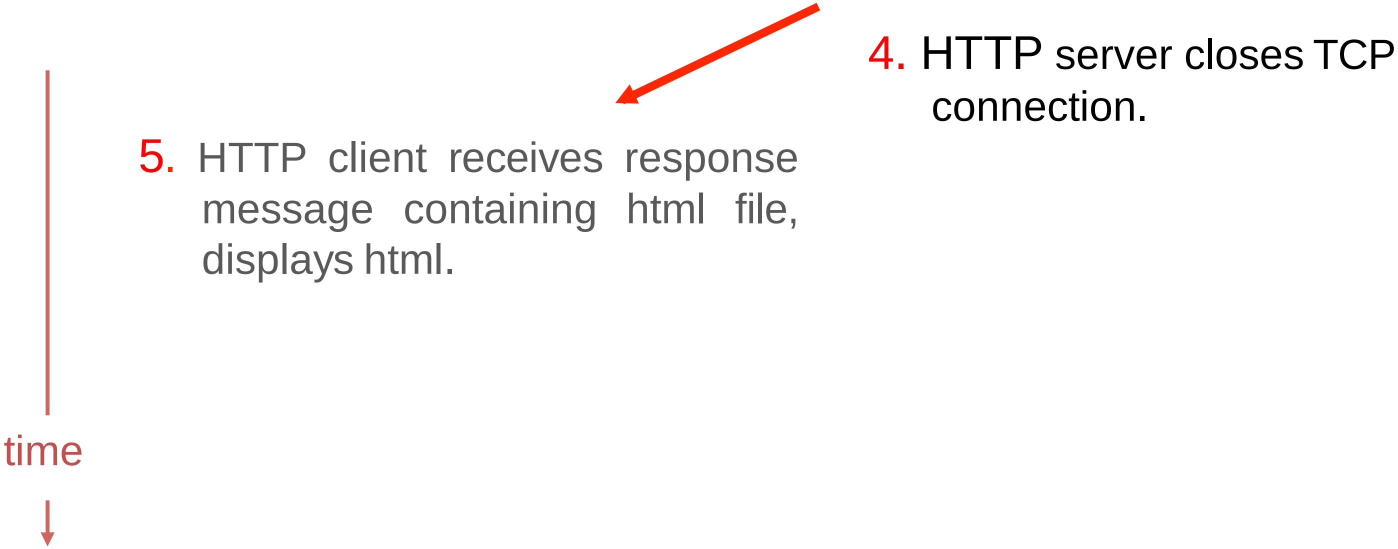
1b. HTTP server at host www.someSchool.edu waiting for TCP connection at port 80. "accepts" connection, notifying client.

2. HTTP client sends **request message** (containing URL) into TCP connection socket. Message indicates that client wants object someDepartment/home.index

3. HTTP server receives request message, forms **response message** containing requested object, and sends message into its socket.

time

Normal HTTP requests (2)



4. HTTP server closes TCP connection.

5. HTTP client receives response message containing html file, displays html.

- HTTP works as a **request-response protocol** between a client and server
- A web browser may be the client, and an application on a computer that hosts a web site may be the server

HTTP request message

FORMAT

method URL version<cr><lf>

header1:value<cr><lf>

header2:value<cr><lf>

…

headerN:value<cr><lf>

<cr><lf>

BODY OF HTTP REQUEST

EXAMPLE

GET /somedir/page.html HTTP/1.1

Host: www.someschool.edu

User-agent: Mozilla/4.0

Connection: close

Accept-language: fr

(extra carriage return, line feed)

HTTP request message – GET

EXAMPLE OF GET METHOD:

```
GET /form.php?username=Joe HTTP/1.1

Host: www.cs.iastate.edu


<CRLF>
```

- The query string (name/value pairs) is sent in the URL of a GET request
- GET passes arguments on URL
- Data limited

- GET is used to request data from a specified resource
- one of the most common HTTP methods
- length restrictions
- only used to **request data** (not modify)
- Data is visible to everyone in the URL (not safe)
- Only ASCII characters allowed

HTTP request message – POST

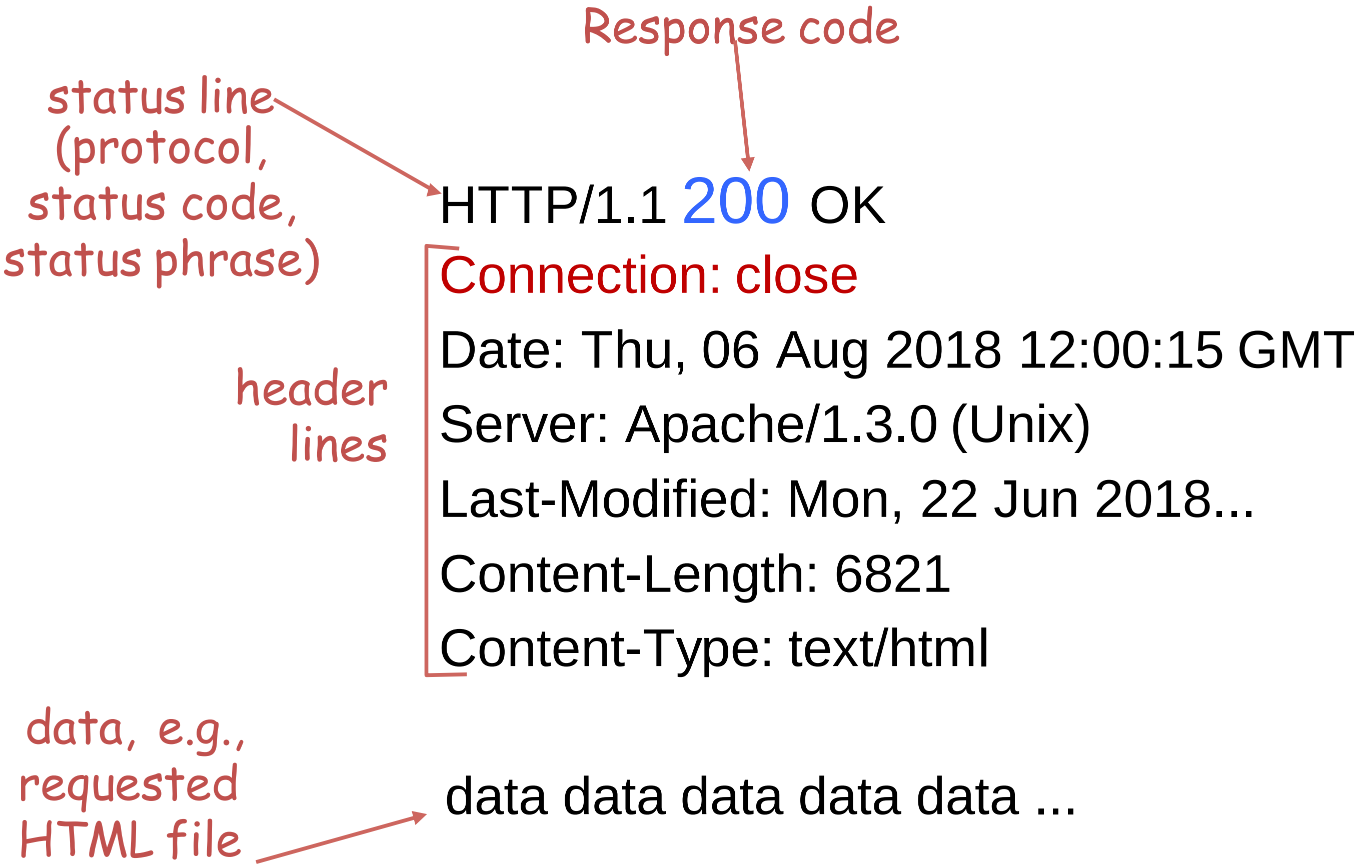
EXAMPLE OF POST METHOD:

```
POST /form.php HTTP/1.1
Host: www.cs.iastate.edu
Content-Length: 12
<CRLF>
username=Joe // This is the body of the request
```

- POST passes arguments in body
- Unlimited data (files) can be sent

- POST is used to send data to a server to **create/update** a resource
- The data sent to the server with POST is stored in the request body of the HTTP request

HTTP response message



Response code

status line (protocol, status code, status phrase)

header lines

data, e.g., requested HTML file

```
HTTP/1.1 200 OK
Connection: close
Date: Thu, 06 Aug 2018 12:00:15 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Mon, 22 Jun 2018...
Content-Length: 6821
Content-Type: text/html

data data data data data ...
```

HTTP response status codes

`200 OK`

- request succeeded, requested object later in this message

`301 Moved Permanently`

- requested object moved, new location specified later in this message (Location:)

`400 Bad Request`

- request message not understood by server

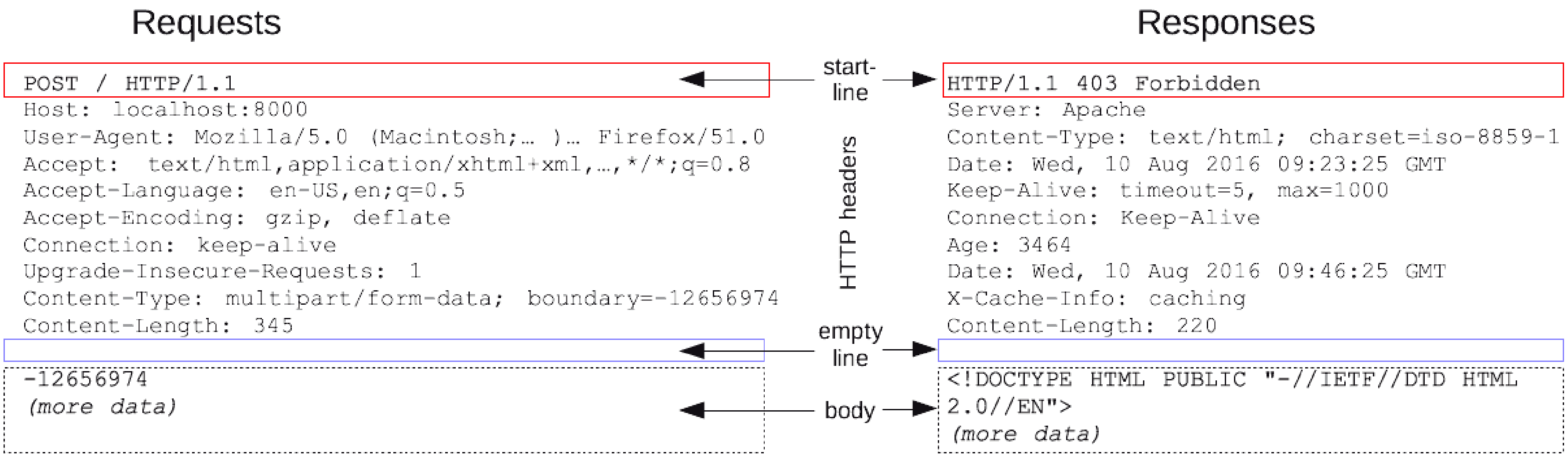
`404 Not Found`

- requested document not found on this server

`505 HTTP Version Not Supported`

HTTP requests / responses

- **Similar structure:**



Requests

```
POST / HTTP/1.1
Host: localhost:8000
User-Agent: Mozilla/5.0 (Macintosh;… )… Firefox/51.0
Accept: text/html,application/xhtml+xml,…,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Content-Type: multipart/form-data; boundary=-12656974
Content-Length: 345

-12656974
(more data)
```

Responses

```
HTTP/1.1 403 Forbidden
Server: Apache
Content-Type: text/html; charset=iso-8859-1
Date: Wed, 10 Aug 2016 09:23:25 GMT
Keep-Alive: timeout=5, max=1000
Connection: Keep-Alive
Age: 3464
Date: Wed, 10 Aug 2016 09:46:25 GMT
X-Cache-Info: caching
Content-Length: 220

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML
2.0//EN">
(more data)
```

Source: https://developer.mozilla.org/

Trying out HTTP (client side)

1. **Telnet to your favorite Web server:**

```
telnet www.iastate.edu 80
```

Opens TCP connection to port 80
(default HTTP server port)

2. **Type in a GET HTTP request:**

```
GET /  HTTP/1.1
```

By typing this in (hit carriage
return twice), you send
this minimal (but complete)
GET request to HTTP server

3. **Type in another GET HTTP request:**

```
GET /  HTTP/1.1
Host: www.iastate.edu
```

Literature – HTTP, HTML, Java Sockets,…

Good resources:

- https://www.w3.org
- **https://www.w3schools.com/**
- http://www.htmldog.com/
- https://www.quackit.com/
- http://www.landofcode.com/
- https://docs.oracle.com/javase/tutorial/networking/sockets/index.html