ITMD 441/541
Web Application Foundations

Week 2

FALL 2023 – AUGUST 28, 2023

Week’s Agenda

- Development Tools
- HTTP Protocol and Request/Response Cycle
- Assignments

Development Tools

Browser Development Tools

- Browser Development Tools
- Built into all major browsers now
- Provide a way to inspect a page, modify CSS, interact with a JavaScript Console, profile, monitor network requests, and more.
- Chrome Dev Tools
- https://developer.chrome.com/docs/devtools/

- Let’s look at some of the options in the browser dev tools

Other helpful apps

- Local HTTP server environment
  - XAMPP
  - MAMP
  - PHP has built in server with php -S command
  - NodeJS package http-server
    - Simple static web server
- HTTP Client to test requests and API connections
  - Available as desktop apps and browser extensions
  - https://chrome.google.com/webstore/detail/talend-api-tester-free-ed/aejoelaoggembcahagimdiliamlcdmfm/related?hl=en

HTTP(S)

THE PROTOCOL THAT POWERS THE WEB

https://developer.mozilla.org/en-us/docs/web/http

HTTP

- Hypertext Transfer Protocol (HTTP)
- First designed by Tim Berners-Lee at CERN
- OSI and TCP/IP application layer protocol for transporting HTML documents and other hypermedia across the internet
- Client-sever protocol that works on a request-response concept
- Designed as a stateless protocol - no state (data) link between two requests on the same connection
  - Addition of HTTP cookies allows stateful sessions
- Simple text-based protocol
- Uses port 80 by default

HTTPS

- Hypertext Transfer Protocol Secure (HTTPS)
- HTTP protocol extension
- First created in 1994 by Netscape Communications for the Netscape Navigator web browser
- HTTP through an encrypted SSL/TLS connection
- Originally used the SSL (secure sockets layer) protocol to add encryption
- SSL evolved into TLS (transport layer security)
- SSL went from version 1 to 3 in a couple years to fix problems and TLS 1.0 was first defined in 1999
- SSL 3 was depreciated in June 2015 and TLS 1.0 and 1.1 were deprecated by all major vendors in March 2020
- TLS 1.3 is the most recent version

HTTPS

- Uses public and private keys to generate and encrypt a short-term session key which is used to encrypt/decrypt the data.
- Certificate authorities and public key certificates are used to verify the owner and validity of a key
- TLS Handshake occurs and the client and server agree on the ciphers and other parameters. Then the client encrypts a random number using the server's public key and they both generate a session key that is used for encryption and decryption of data
- Regular HTTP protocol is then sent through the encrypted connection
- Uses port 443 by default

HTTP/0.9

- Tim Berners-Lee’s first documented version of HTTP in 1991
- https://www.w3.org/Protocols/HTTP/AsImplemented.html
- Initially didn’t have a version number, later called HTTP/0.9
- Only defined a single HTTP method, GET
- One-line protocol
  - GET followed by resource path
  - GET /index.html
- Response was the contents of the file

HTTP/1.0

- In 1995 Dave Raggett led an HTTP working group to expand the protocol by adding additional methods and headers and formalize all the draft specifications.
- This became HTTP V1.0 in 1996. It defined 3 methods GET, HEAD, and POST
- Appended HTTP version info to the end of the first line of the request
- Status code was added to the first line of the response
- Added headers for extensibility
- Content-Type header allowed for files other than HTML

HTTP/1.1

- First standardized version of HTTP
- HTTP 1.1 was released in January 1997 and added 5 methods: OPTIONS, PUT, DELETE, TRACE, and CONNECT.
- It made the HOST header required.
- Allowed a connection to be reused to save time.
- Content negotiation introduced.
- In 2007 the HTTP working group formed to clarify parts of the HTTP 1.1 specification and in June 2014 released an updated 6-part HTTP 1.1 specification.

HTTP/2

- Protocol to increase performance standardized in May 2015
- Built on Google’s experimental protocol SPDY in 2010’s
- Binary protocol versus a text protocol.
- Multiplexed protocol. Parallel requests over same connection.
- Adds data compression for HTTP headers and server push
- High-level compatibility with HTTP/1.1
  - methods, status codes, headers, URIs
  - Websites and applications did not require changes which lead to fast adoption
- https://en.wikipedia.org/wiki/HTTP/2
- https://web.dev/performance-http2/

HTTP/3

- Third major version of HTTP
- Standard published on June 6, 2022
- Semantically like previous versions using same request methods, status codes, and message fields (headers)
- Major difference is uses QUIC (transport layer protocol) over UDP (User Datagram Protocol) instead of TCP
- Can be 3x faster than HTTP/1.1 in some instances
- https://en.wikipedia.org/wiki/HTTP/3

HTTP Request/Response

- Individual requests are sent by a client to a server
- We call the client a **user agent** and it is typically a web browser, but can be any program that understands HTTP like a search engine web crawler or CLI like curl
- The server handles the **request**, processes it, and provides a **response**
- The request is text formatted (pre http/2) data in a specific format that can be as short as two lines of text.
  - Method and the resource being requested
  - Host header saying the domain it if being sent to, required as of 1.1
  - Each line ends with a carriage return (cr) followed by a line feed (lf) and the entire request ends with a double newline
  - A Request body may follow the headers if the request is sending data
- Response follows a similar format

What is a URL?

- URL or Uniform Resource Locator
- Just an address for a unique resource on the web
- URL is made of of different parts
- https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

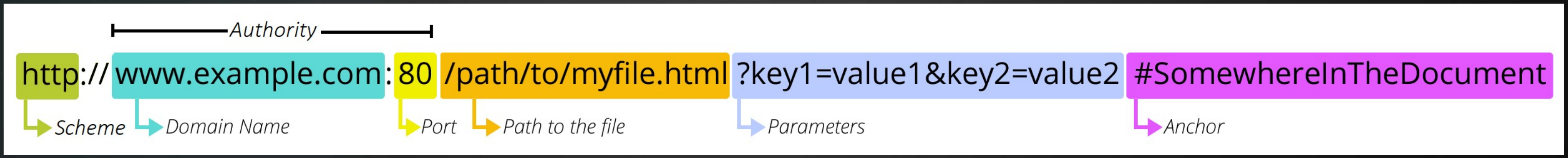
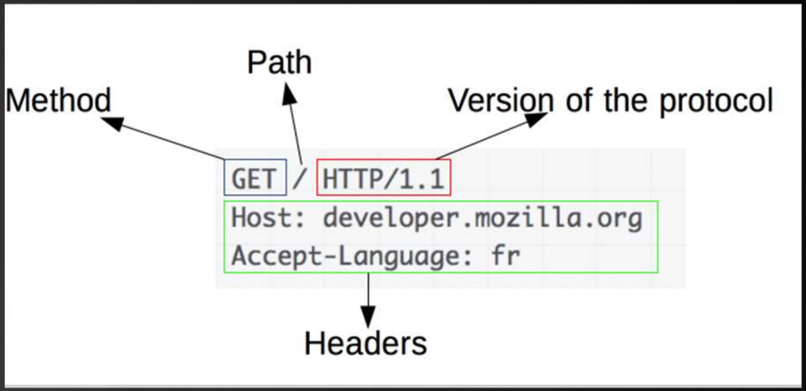

Authority
http://www.example.com:80/path/to/myfile.html?key1=value1&key2=value2#SomewhereInTheDocument
Scheme
Domain Name
Port
Path to the file
Parameters
Anchor

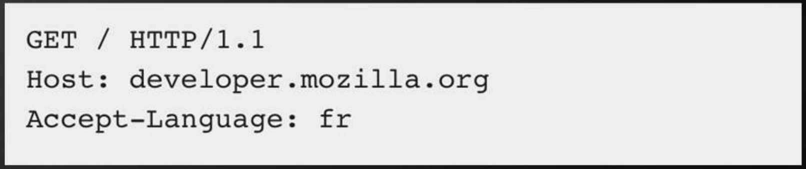
Image from https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

HTTP Request

- Requests contain:
- HTTP method for the operation the client is trying to perform.
- Path of the resource, no protocol, domain, or port
- HTTP protocol version
- Headers for additional information (optional)
- Request body for some methods like POST



```
GET / HTTP/1.1
Host: developer.mozilla.org
Accept-Language: fr
```

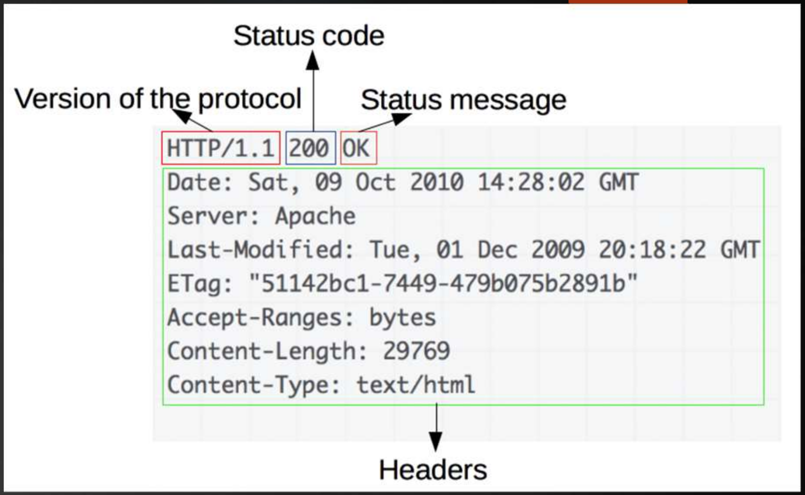


```
GET / HTTP/1.1
Host: developer.mozilla.org
Accept-Language: fr
```

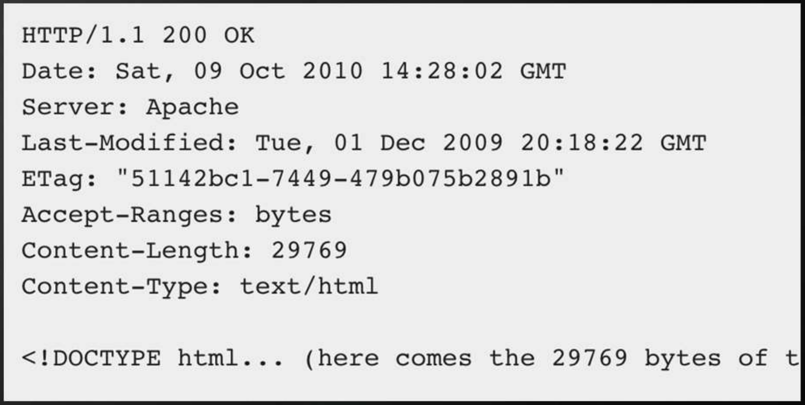
Images from https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

HTTP Response

- Response contains:
- HTTP protocol version
- Status code representing if the request was successful or not and the reason.
- Status message. A short description of the status code.
- Headers for additional information (optional)
- Response body that contains the fetched resource if there is content returning to client.



```
HTTP/1.1 200 OK
Date: Sat, 09 Oct 2010 14:28:02 GMT
Server: Apache
Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
ETag: "51142bc1-7449-479b075b2891b"
Accept-Ranges: bytes
Content-Length: 29769
Content-Type: text/html
```



```
HTTP/1.1 200 OK
Date: Sat, 09 Oct 2010 14:28:02 GMT
Server: Apache
Last-Modified: Tue, 01 Dec 2009 20:18:22 GMT
ETag: "51142bc1-7449-479b075b2891b"
Accept-Ranges: bytes
Content-Length: 29769
Content-Type: text/html

<!DOCTYPE html... (here comes the 29769 bytes of t
```

Images from https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview

HTTP Flow

- Client opens a TCP connection to the server
  - May be a new or existing connection
  - May open several connections
- Sends an HTTP request message
- Server processes request and sends response message to client
- Client reads response send by server
- Client closes or reuses the connection for other requests
- Client sends request for each needed resource on page

HTTP Methods

- HTTP Methods
- GET, POST, PUT, DELETE, HEAD, CONNECT, OPTIONS, TRACE, PATCH
- GET and POST are most common

- GET
  - Request for a specified resource
  - Should only be used to retrieve data
  - Any data is sent as part of URL using query parameters
  - http://www.domain.com/?q=232&name=joe
  - Data q=232 and name=joe is available to target resources

HTTP Methods

- POST
  - Submits data to a specified resource and often causes changes in data state on the server
  - Almost always used to send form data
  - Any variable or form data is sent in the request body and not appended to the URL
  - Type of data in request body would be indicated by the Content-Type header

HTTP Methods

- PUT
  - Replaces data or resource on the server with request content
  - Not used in HTML forms
- DELETE
  - Delete specified resource
  - Not used in HTML forms
- Used more with AJAX, frameworks, and non-browser HTTP requests

HTTP Methods

- HEAD
  - Just like a GET request but only returns headers and not the body
- OPTIONS
  - Requests communications options from server
- CONNECT
  - Starts a two-way communication with a resource
- TRACE
  - Performs a loop-back test for debugging
- PATCH
  - Partial modifications to a resource

HTTP Status Codes

- HTTP Status Codes
- 3-digit numbers grouped into 5 groups by first digit
- 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 – Successful
- 200 OK
  - The request succeeded. Exact details depend on method.
- 201 Created
  - The request succeeded and new resource was created.
  - Typically sent as a result of POST or PUT

HTTP Status Codes

- 3xx – Redirection
- 301 Moved Permanently
  - URL of requested resource has permanently moved, and the new URL is given in the response. Search engines should update and use new URL.
- 302 Found
  - URL of requested resource has moved temporarily, and the new URL is given in the response. May change again in the future so the same URL should be used.
- 304 Not Modified
  - Requested resource has not been modified. Used to help caching.

HTTP Status Codes

- 4xx Client error
- 400 Bad request
  - Incorrect request syntax
- 401 Unauthorized
  - Client not allowed access to resource
  - May change if client retries with authentication
- 403 Forbidden
  - Client not allowed access to resource
  - Client identity is known to server and further authentication will not help
- 404 Not found – Dead link

HTTP Status Codes

- 5xx Server error
- 500 Internal Server Error
  - Something went wrong inside the server, or it doesn’t know how to process
- 501 Not Implemented
  - The request is not supported by the server
- 503 Service Unavailable
  - Server not able to handle the request
  - Usually happens when a server is overloaded or down

HTTP Cookies

- A HTTP Cookie is a small amount of text data that can be sent from the server in a response and stored in the client.
- The browser can then send the data back to the same server on future requests
- This allows the server to know if two requests came from the same client
- This is what allows our stateless HTTP protocol to be able to remember information like keeping a user logged in.
- Cookies are primarily used for session management, personalization, and tracking
- Cookies are set in a response using a Set-Cookie header and returned to the server by the client automatically in a request using the Cookie header
- Should not be used to store general data in the client, there are modern APIs
- https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies

HTTP(S) Resources

- https://developer.mozilla.org/en-US/docs/Web/HTTP
- https://www.semrush.com/blog/what-is-https/
- https://www.cloudflare.com/learning/ssl/what-is-https/
- https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/
- https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:web-protocols/a/hypertext-transfer-protocol-http
- https://hpbn.co/

Assignments

Reading/Assignments

- Quiz 1 – Week 1 Content – Due end of day Friday, September 1 before midnight
- Read about HTTP on MDN
- https://developer.mozilla.org/en-US/docs/Web/HTTP
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Evolution_of_HTTP
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Session
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods
    - https://developer.mozilla.org/en-US/docs/Web/HTTP/Status
- Quiz 2 – Week 2 Content – Assigned this weekend