CS 4350: Fundamentals of Software Engineering
CS 5500: Foundations of Software Engineering

Lesson 3.2: HTTP Basics

Jon Bell, John Boyland, Mitch Wand

Khoury College of Computer Sciences

Learning Objectives for this Lesson

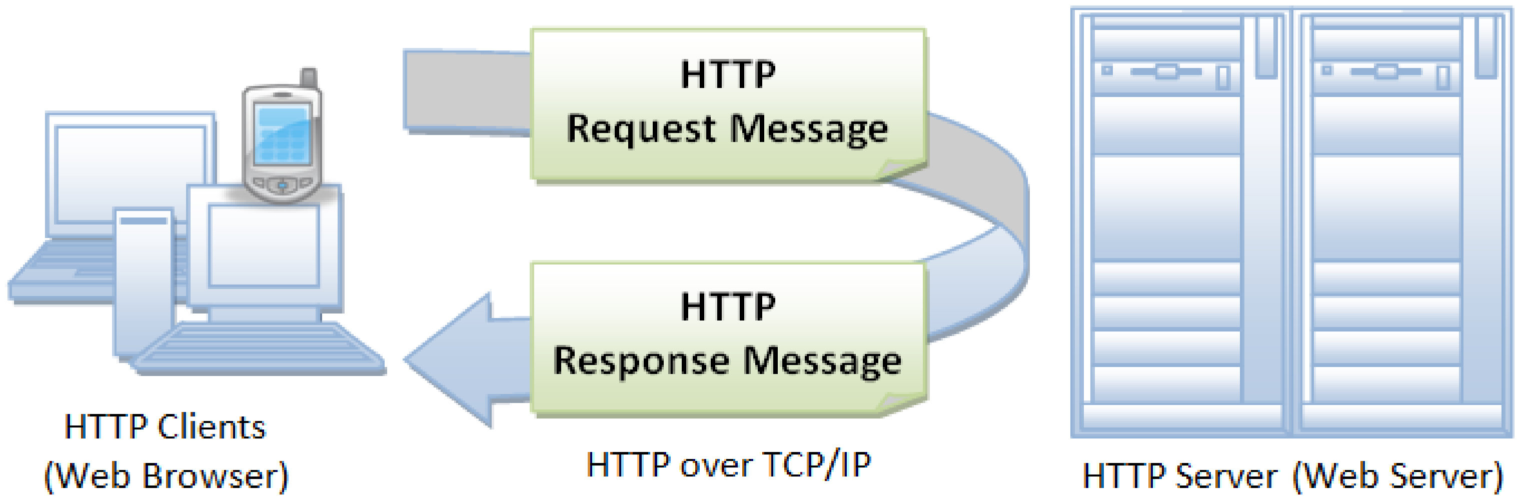
- By the end of this lesson you should be able to:
  - Explain the basic structure of the HTTP protocol
  - Define the following terms in the context of HTTP:
    - client, server
    - request, response
    - message header
    - message body
  - Explain the basic flow of an HTTP request
  - Explain what is meant by the following:
    - HTTP is stateless but not sessionless

What is HTTP?

- HTTP stands for "HyperText Transfer Protocol"
- Originally developed (1989-1991) by Tim Berners-Lee as a protocol for transmitting web pages.
- Has evolved into a general-purpose character-based protocol for communicating on the web.

HTTP is Asymmetric

- It distinguishes between client and server.
- The client initiates a request, and the server replies by sending a response.

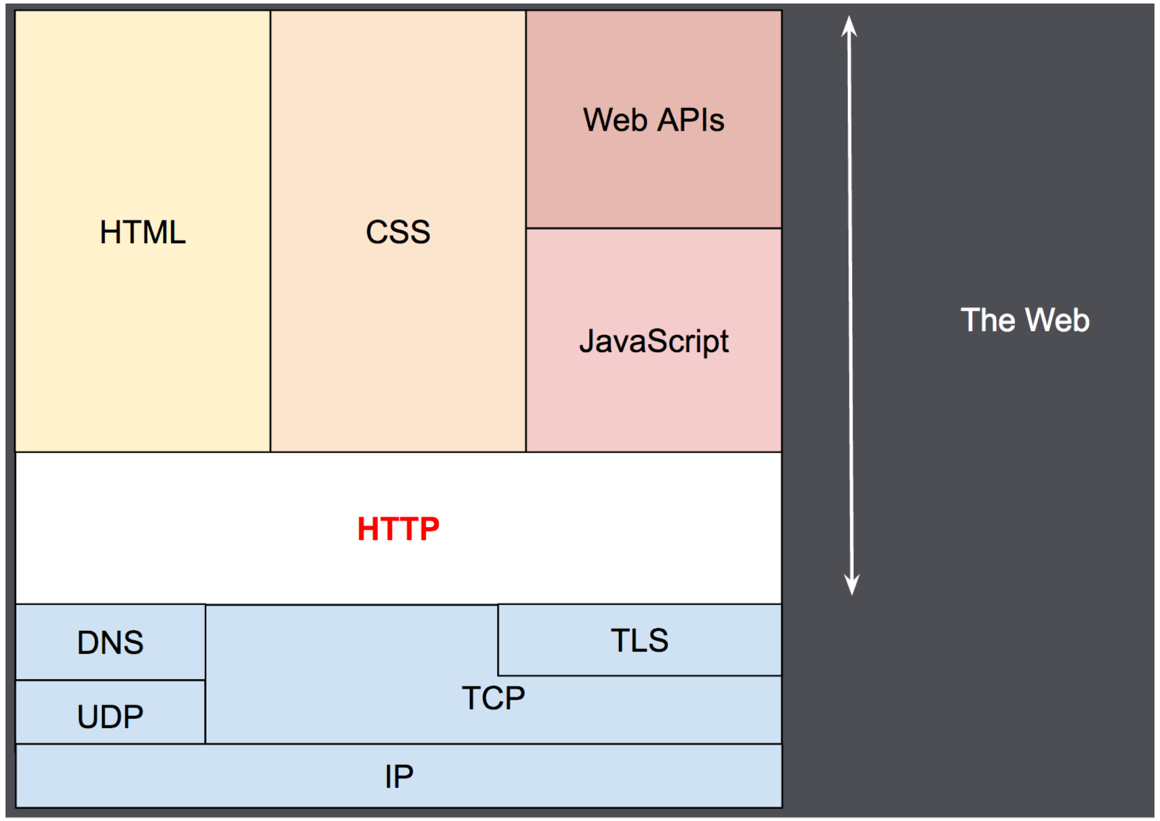


https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html

HTTP is Stateless

- Each request/response pair is independent.
- If the client intends the request to be part of a session, then the request must include all of the data needed to allow the server to resume the session at that point.
  - we'll see later how this data can be included in the request.
- We say "Http is stateless but not sessionless"

HTTP is an Application-Level Protocol



Usually sits on top of TCP/IP, but other transport layers are possible.

https://developer.mozilla.org/en-us/docs/Web/HTTP/Overview

HTTP is an Application-Level Protocol

- Originally designed for transmitting HTML data, but can be used for almost anything.
- It's up to the client and the server to decide on what the data is and how it is to be interpreted
- The interpretation may be arbitrarily complex, but we'll learn about a useful class of simple protocols, called REST.

Remember Design Principle #2: Define Your Data!

HTTP is Extensible

- A request comes with headers, which define the kind of data the client is sending and the kind of data it expects (or is willing to) receive.
- Similarly, a response comes with headers, which indicate whether the request was successful or not, and the format or formats in which the reply data is transmitted.
- Headers may also include many other kinds of metadata.
- The set of possible headers has grown tremendously over the years.

Oops! The version in the video says "server". It's the client who sends the request, of course.

HTTP Flow Step 1: Client opens a TCP Connection

- This is the connection over which data will flow
- Client can create a fresh connection or re-use an existing one.

HTTP Step 2: Client sends a Request

- Consists of:
    - a method
        - one of a few simple verbs
    - a path
        - a URL
    - a version
    - headers
    - maybe a body
        - separated from the headers by a blank line

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html

HTTP Methods (Verbs)

- Each request comes with a method.
- There are just a few of these. Here are the ones that you should know:
- GET: requests a representation of a resource
  - the path is the name of the resource
- POST: requests the server to create a resource
  - the path is the name of the resource
  - there are several ways in which the value of the new resource can be transmitted (more later)
- PUT: requests the server to change the value of the resource to the given value
- DELETE: requests the server to delete the resource

GET is what a browser does.

HTTP Step 3: Server interprets the Request

```
GET /docs/index.html HTTP/1.1
Host: www.nowhere123.com
Accept: image/gif, image/jpeg, */*
Accept-Language: en-us
Accept-Encoding: gzip, deflate
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
(blank line)
```

- This request probably started out as http://www.nowhere123.com/docs/index.html
- www.nowhere123.com identifies the host (the server's location)
- the rest of the request is the path, here /docs/index.html
- this might be a path in the server's file system,
- OR it could be anything at all—
- it's entirely up to the server to interpret the path

We'll see later how these paths are interpreted in REST protocols.

HTTP Step 4: Server sends a Response

- Consists of
  - a version
  - a status code
  - a status message
  - some headers
  - maybe a body

```
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 the requested web page)
```

https://developer.mozilla.org/en-us/docs/Web/HTTP/Overview
(NOT a response to the preceding request!)

HTTP Step 5:
The client closes or reuses the connection

Review: Learning Objectives for this Lesson

- You should now be able to:
  - Explain the basic structure of the HTTP protocol
  - Define the following terms in the context of HTTP:
    - client, server
    - request, response
    - message header
    - message body
  - List the steps in the basic flow of an HTTP request
  - Explain what is meant by the following:
    - HTTP is stateless but not sessionless

Next steps…

- In the next lesson, we will learn about REST, a philosophy for designing application-level protocols on top of http.