Recitation 5: HTTP Pseudo Streaming

MARCH 26, 2021

Joel Miller

Agenda

1. HTTP Explained
2. Project 3
3. Live Demo!

HTTP Explained

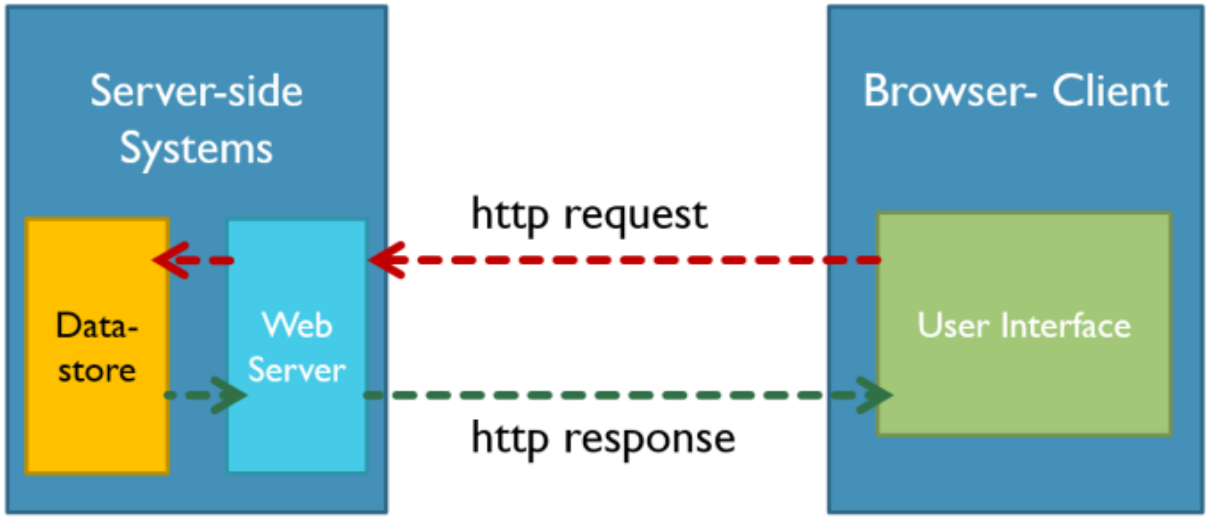
What is HTTP?

- **H**yper**T**ext **T**ransfer **P**rotocol is an application-layer protocol that enables the transfer of hypermedia (the linkable format of data on the World Wide Web).
- Used by almost all modern websites to transfer content from their computer to yours

HTTP Explained

The HTTP Model

- Client-server based model
- Clients make requests, servers send responses



HTTP Explained

What is a URI?

- **U**niversal **R**esource **I**dentifier identifies a particular *resource*
- A resource is an abstract "thing" that is pointed to by a URI
  - Could be a json, could be a file, could be a list of your favorite pasta recipes…
- Clients only ever sees a representation of the desired resource
- Examples: www.google.com/myaccount, https://www.cmu.edu/academics/index.html

HTTP Explained

What is a URL?

- **U**niversal **R**esource **L**ocator is a special type of URI that tells you *how* to find a resource
- Specifies protocol information, host information, resource identification information, etc.
- Example:
  - http://www.google.com/account is a URL
  - www.google.com/account is a URI since it fails to specify protocol

HTTP Explained

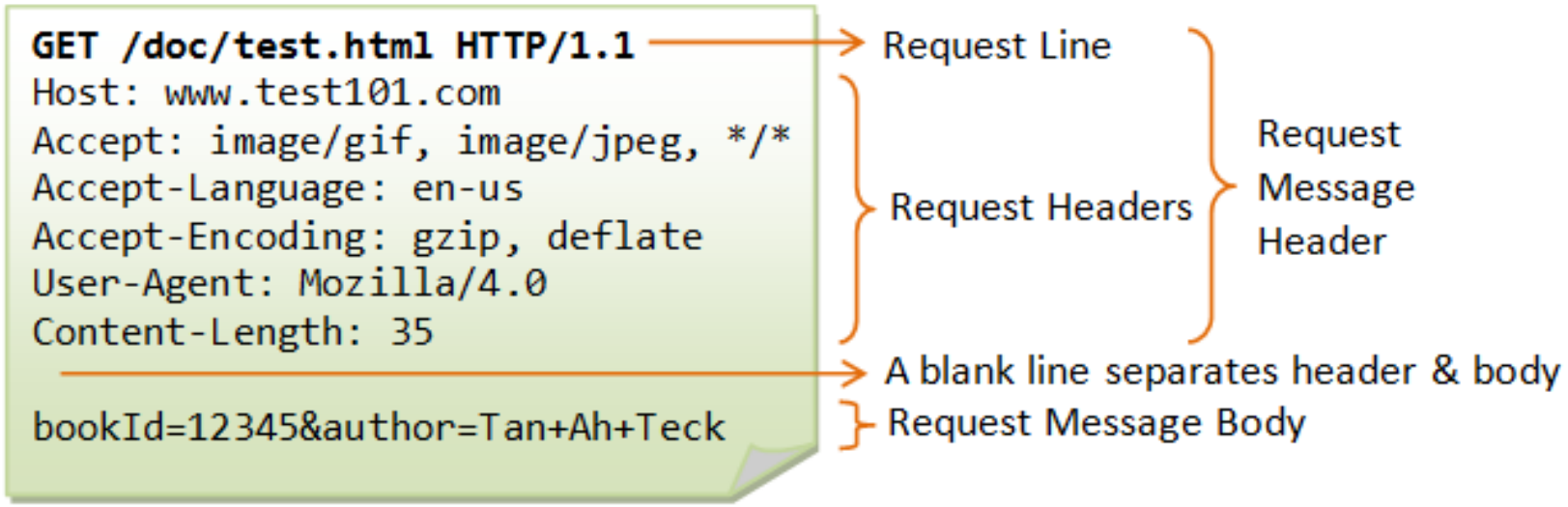
How do I access resources?

- HTTP requests have different *methods* for interacting with resources:
    - GET is used to request a representation of a resource
    - POST is used to submit data to a resource
    - PUT is used to overwrite all representations of the resource
    - DELETE is used to delete the resource

HTTP Explained

HTTP Request Format

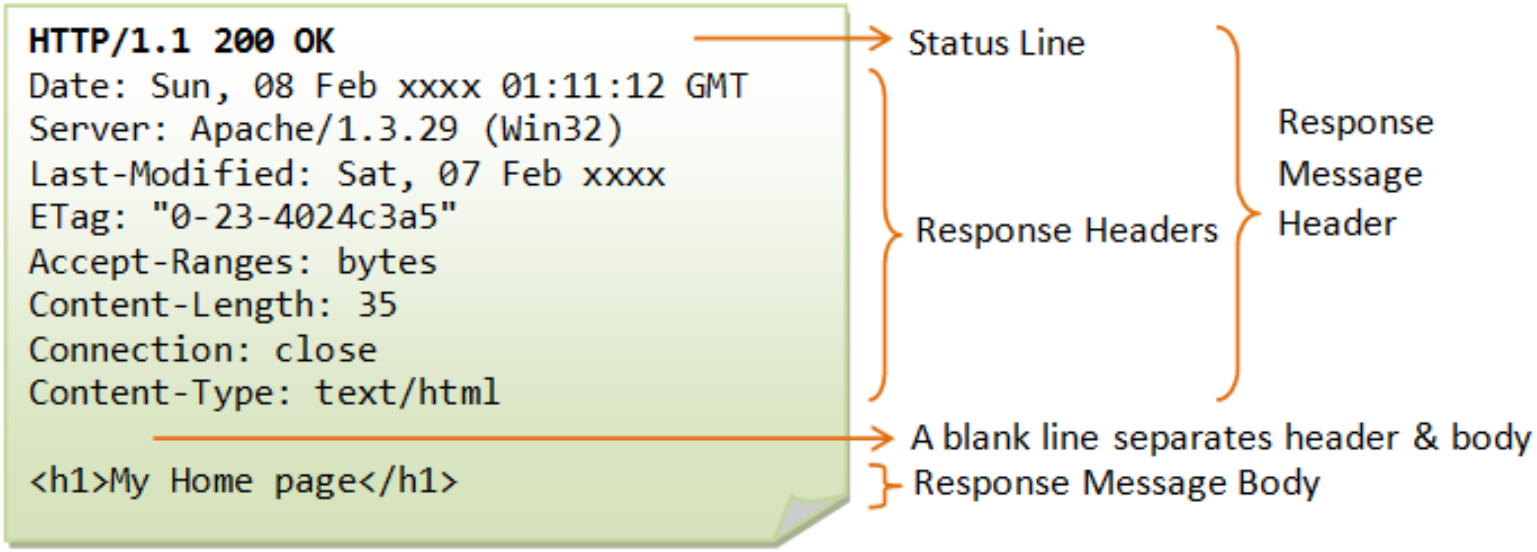
- It's all just one big string!



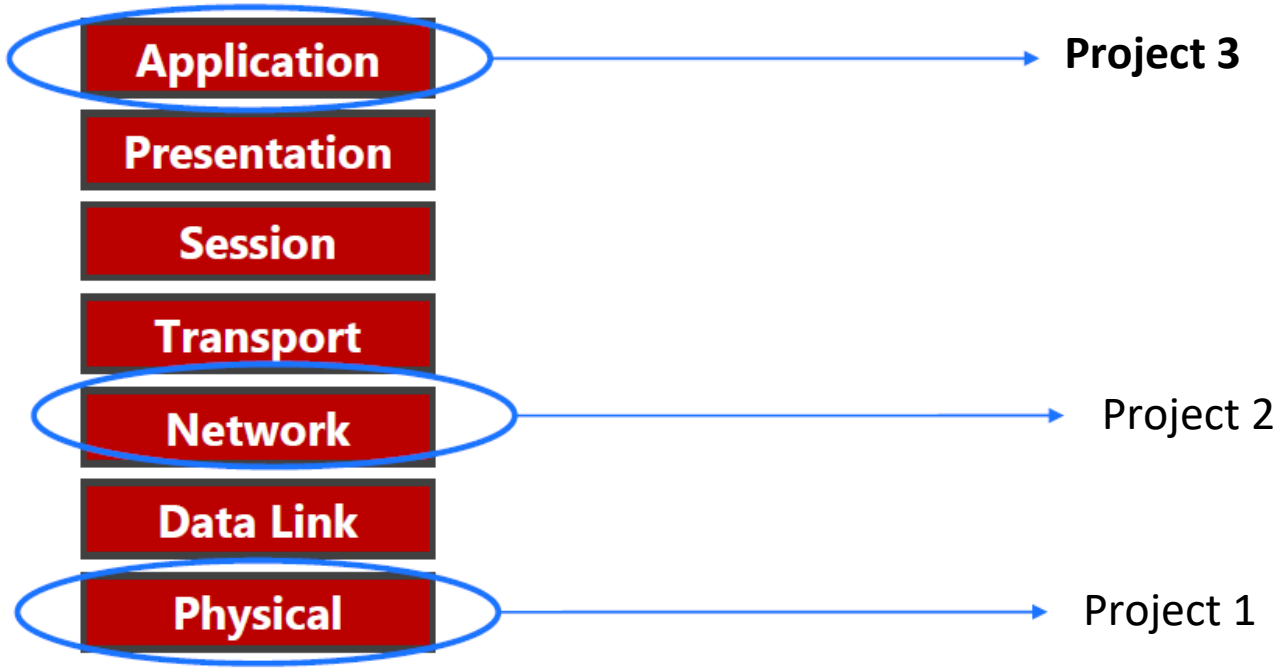
HTTP Explained

HTTP Response Format

- Wow, another big string!



Project 3


Application
Presentation
Session
Transport
Network
Data Link
Physical
Project 3
Project 2
Project 1

Project 3

Your mission (should you choose to accept it)

- Build an HTTP (web) server that can deliver content to a real browser!
- Your server, `vodserver`, should be able to deliver many different file types including streaming video

Project 3

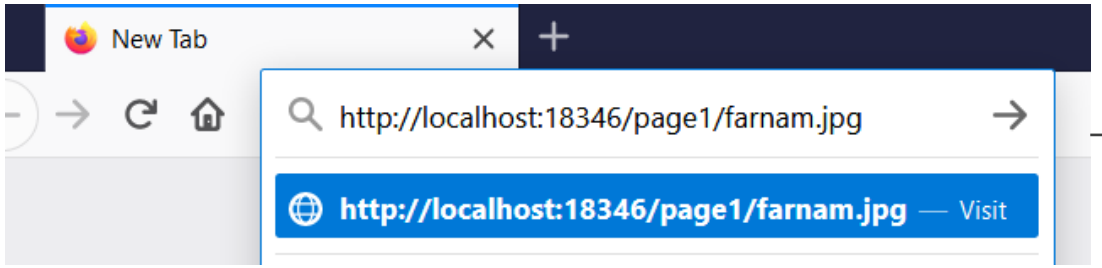
What content are we delivering?

- Your web server should deliver files specified by the user via the URI
- The files to deliver will be located in the `content` folder relative to the server executable file

Project 3

Example User Request

Client types in:



Firefox generates:

```
GET /page1/farnam.jpg HTTP/1.1
Host: localhost
… (more headers) …
/r/n/r/n
```

To your server

Project 3

Example Server Response

Your server receives & parses:

```
GET /page1/farnam.jpg HTTP/1.1
Host: localhost
… (more headers) …
/r/n/r/n
```

Server retrieves resource

Your server generates:

```
HTTP/1.1 200 OK
Date: Fri, 26 Mar 2021 06:09:42 GMT
… (more headers) …
/r/n/r/n
<Image data from content/page1/farnam.jpg>
```

To the client

Project 3

Video Streaming

- Client will ask for “large” video file
- Server replies with 206 Partial Content, sends part of the video
- Client will buffer video until user reaches the end of the buffer or seeks new location, then requests a range of bytes for the video file
- Server replies with the requested bytes
- Repeat 3rd bullet point until end of video or end of user’s attention span

Project 3

Advanced 18-741 Requirement

- Should handle up to 5000 concurrent clients
- Use ab – Apache HTTP server benchmarking tool (already installed on ECE cluster machines)
- Bonus for 18-441 students!

Live Demo!

Server should handle:

- Plain text (.txt)
- Web pages (.html, .css, .js)
- Images (.jpg, .png, .gif)
- Audio (.ogg)
- Video (.mp4, .webm)

Closing Comments

- Project 3 is due **11:59pm EDT April 25, 2021**
- **Start early!** It will take some time to figure out how to interact with the browser
- Download the latest handout from Canvas (updated just last night)
- We will be using Firefox for testing (see handout for version)
- For browser testing, run your `vodserver` on the ECE cluster and attempt to connect using Firefox on your own computer (Firefox over SSH is painfully slow).
- Don't submit executables! Your grade will be sad ☹
- Make sure to include the `Connection: Keep-Alive` header

Q&A

- Feel free to ask questions on Piazza or in OH later
- OH is mostly empty at the beginning of projects, so make good use of them by starting early!

Extra Resources

- Ab tool documentation: http://httpd.apache.org/docs/2.0/programs/ab.html
- HTTP basics explained: https://www3.ntu.edu.sg/home/ehchua/programming/webprogramming/HTTP_Basics.html
- HTTP resources: https://devblast.com/b/what-are-http-resources