CS144 Notes: HTTP

Basic interaction

- **Example:** [http://www.youtube.com](http://www.youtube.com)
  - Q: what is going on behind the scene?
    - Q: What entities are involved in this interaction?

- Q: What is the role of each entity?
  - Q: What runs on server? client? network?
    Who keeps track of what is being done?

- Q: There are many Web servers on the Internet. How can the Web browser reach and communicate with the YouTube server?

- Q: Many things are exchanged over Internet. Email, instant messaging, file transfer, etc. How does the server know that this "client" wants a "Web page"?

- Q: Only bytes are transferred. How do they communicate rich, dynamic multimedia content?

- **TCP/IP, http, html**
  - TCP/IP (transmission control protocol and internet protocol)
    * internet routing and transportation protocol
  - http (hypertext transportation protocol)
    * communication protocol between web servers and web clients
  - html (hypertext markup language)
    * page formatting and linking standard

**HTTP**

- HTTP/1.1 most popular (HTTP/2 is the most recent)
- Request & response
• Stateless: every request is handled independently from others
  - Q: what are pros/cons of stateless protocol?

• message = request/status line + header + body
  - http request
    * the bare minimum HTTP request -- can be issued through telnet
      GET / HTTP/1.0

    * e.g.
      ```
      GET /cs144/examples/form.html HTTP/1.1 <- request line
      Host: oak.cs.ucla.edu <- beginning of header
      User-Agent: Mozilla/5.0 ...
      Referrer: http://oak.cs.ucla.edu/cs144/
      Accept:text/xml,text/html;q=0.9,text/plain;q=0.8,image/png,*/*;q=0.5
      Accept-Language: en-us,en;q=0.5
      Accept-Encoding: gzip, deflate
      Accept-Charset: ISO-8859-1,utf-8;q=0.7,*,*;q=0.7
      Keep-Alive: 300
      Connection: keep-alive
      Cookie: __utmz=125574670.1174236576.14.14...
      ```
      <--- end of header

  - request line: the actual request
    * more on the "GET" method later

  - header: additional information for the request
    * Host: the name of the web server
      - Q: why do we need the "Host:" field? Aren’t we already contacting it?

    * User-Agent: information on the client software
    * Referrer: The page linking to the requested page.
      - Q: how can it be used? Can the server reconstruct the user's click path?
* Accept ... : what media/content is acceptable to the client q=... specifies how much the type is "preferred"

* Keep-Alive, Connection: in case we want to make multiple requests through one connection
  - Q: why do we want to make multiple requests per connection?

* Cookie: more on this later

- http response
  - e.g.

Apache

HTTP/1.1 200 OK  
Date: Wed, 04 Apr 2007 03:20:33 GMT  
Server: Apache

2xx: Success - The action was successfully received, understood, and accepted
3xx: Redirection - Further action must be taken in order to complete the request
4xx: Client Error - The request contains bad syntax or cannot be fulfilled
5xx: Server Error - The server failed to fulfill an apparently valid request

- ETag: a unique tag that is the same only if the body is the same
  - Q: when will it be useful?

- Content-Length: length of the body
- Content-Type: the type of the content html, flash, pdf, etc.

- favicon (favorite icon. small icon next to the URL)
  - default: /favicon.ico of the site
  - can be customized for every page by adding the following to

  "Example page"

- Status line:
HTTP/2
- HTTP standard approved on Feb 17, 2015
- Backward-compatible extension to HTTP/1.1
  * Data compression of HTTP headers
  * Parallel loading of page elements over a single TCP connection
  * Server push technologies