**WDI Notes - W02 D01**

SERVER

* download
* upload

CLIENT

Clients:

* Chrome browser {assembles your page from HTML, CSS, JS file – also reads image files and a few other types}
  + Hyper(t)ext Transfer Protocol (http)
    - The protocol the clients follow
  + Other clients such as file share websites may use File Transfer Protocol (FTP)

Requests and Responses:

Client request:

GET <http://www.google.com> /index.html

(http verb) (host name) (URI Uniform Request Identificat)

Server response: *#packages the following:*

Status: Ok 200

Type: HTML

Version: Google super file server 23

Body : <body type html>

Chrome: *#unbundles (mostly abstraction)*

What does the application layer do?

* Makes sure that the other party is identified and can be reached
* If appropriate, authenticates either the message sender or receiver or both
* Makes sure that necessary communication resources exist (for example, is there a modem in the sender's computer?)
* Ensures agreement at both ends about error recovery procedures, data integrity, and privacy
* Determines protocol and data syntax rules at the application level

Application Layer ^

Transport Layer (TCP) { ABSTRACTION

Internet Layer (IP) { METAL 🡪 physical connection

Network (PPP) ^

Separation of Concerns

Server

* cares about:
  + data storage
  + request validation
  + routing
* doesn’t care about:
  + user interface
  + user context
  + session storage

Client

* cares about:
  + user interface
  + portable code #universal portability 🡪 can run anywhere
  + session storage *#will matter later in class*
* doesn’t care about:
  + long-term data storage

What does it mean to be a full-stack developer?

* Can write both server-side and client–side code.

RESTful architecture (REST: representational state transfer)

URIs/Resource (naming conventions):

* + Collection URIs
    - Myhost.com/resources
    - Ex. Myhost.com/users
      * Might give you back a list of users
  + Element URIs
    - Myhost.com/resources/num12
    - Ex. Myhost.com/users10
      * Might give you back a specific user

URI verbs:

* + GET
    - Collection URI: List the members in the collection
    - Element URI: Retrieve a single member in the collection
  + PUT
    - Collection URI: Replace the collection with
    - Element URI: Replace the member with
      * *#typically creates a new box (parameters)*
  + POST
    - Collection URI: Creates one or more new member(s) in the collection
      * *#typically creates a new box (parameters)*
    - Element URI: X
  + DELETE
    - Collection URI: Delete one or new members of the collection
    - Element URI: Deletes a single member of a collection

Statelessness

Servers are priced on bandwidth (Bandwidth 🡪 speed vs. memory)

Localhost: reserved domain name for your own computer

# HTTP VERB + URI PATTERN + BLOCK OF CODE  
get “/”, do

“Hello World!”

end