Write your name below and indicate your role,

Project Manager (PM), Recorder (R)

Name ____ Role ________

Name ____ Role ________

HTTP and Abstraction

Your Tasks

- ☐ Understand the role of *abstraction* as it relates to the Internet
- ☐ Watch the video The Internet: HTML and HTTP
- ☐ Describe how HTTP is used to send and receive data
- ☐ See HTTP in action
- ☐ Define key vocabulary
- ☐ Receive credit for the group portion of this lab

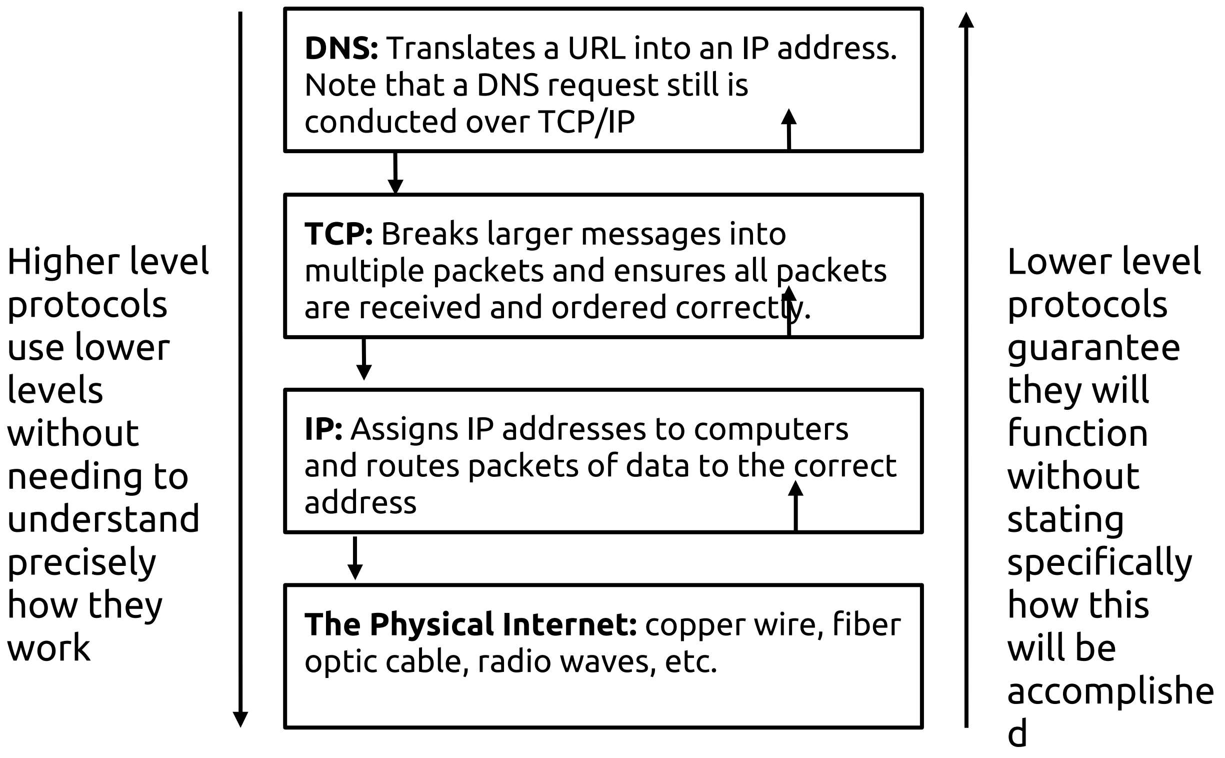
☐ Understand the role of *abstraction* as it relates to the Internet

HTTP like DNS is an ASCII-text based protocol - it's just two computers sending text messages to each other. What makes it a protocol are the rules of the "conversation" the two machines are having. In the case of HTTP, it is a call-and-response protocol for a client/server relationship, where a client requests a web page or other content (image, sound, video, etc.) from a server. The server looks for it and sends it back.

HTTP is a "high level" protocol that sits on top of all the other protocols and internet systems we've learned about so far. That text message conversation between the computers is broken up into TCP/IP packets, and all the data gets sent as bits over wires and airwaves, taking different paths, until it gets interpreted and reassembled at the end.

We often talk about how the Internet works in "layers" and this is a perfect example of ***abstraction*** on the Internet, as one layer makes use of the functionality provided by the layer below it, without worrying about the details of how this functionality is achieved. HTTP doesn't have to worry about anything other than the text protocol upon which HTTP works. The network software and devices on your and others' computers handle looking up addresses, breaking down data, packeting, routing, transmission and interpretation and reassembly. It's really amazing.

Abstraction plays a key role in the relationship between the layers of the Internet, as higher levels make use of the functionality provided by lower levels, without worrying about how they function.


Higher level protocols use lower levels without needing to understand precisely how they work
DNS: Translates a URL into an IP address. Note that a DNS request still is conducted over TCP/IP
TCP: Breaks larger messages into multiple packets and ensures all packets are received and ordered correctly.
IP: Assigns IP addresses to computers and routes packets of data to the correct address
The Physical Internet: copper wire, fiber optic cable, radio waves, etc.
Lower level protocols guarantee they will function without stating specifically how this will be accomplishe d

Define the term abstraction. Explain the role of abstraction on the Internet

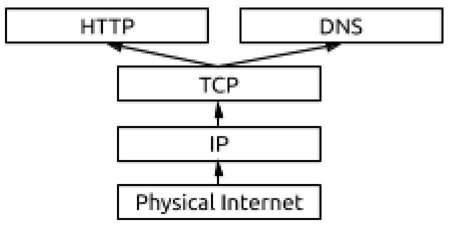
◻ Watch the video The Internet: HTML and HTTP

Navigate to the following link and watch video The Internet: HTML and HTTP

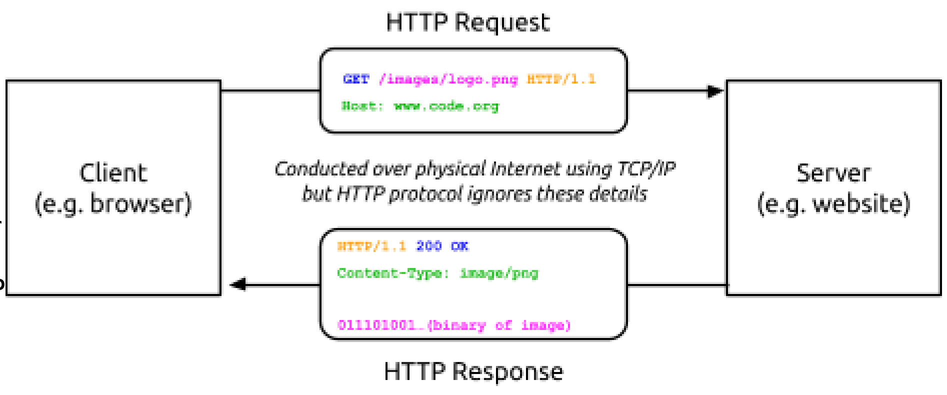
https://www.youtube.com/watch?v=kBXQZMmiA4s&feature=youtu.be

☐ Describe how HTTP (HyperText Transfer Protocol) is used to send and receive data

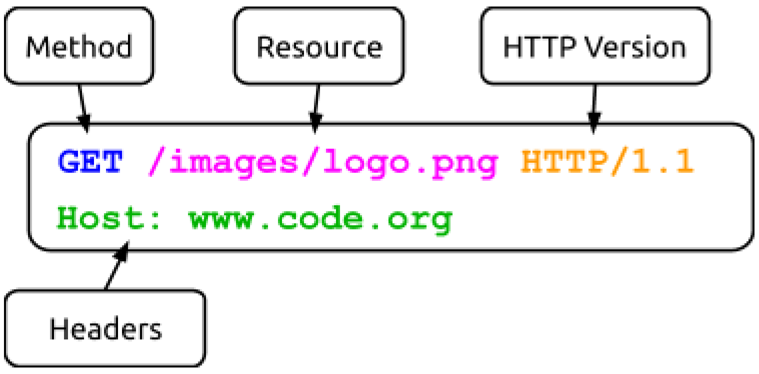
HTTP is a high level protocol, that defines how users of the Internet (clients) request and receive data like web pages, images, video, audio, and files from the servers containing them. A client will send a request to the server with an identifier for a desired piece of data, and the server will attempt to respond to the request, typically by returning the information requested.



HTTP Requests
When you type a URL in your browser, your computer (the client) needs to "ask" the server that is storing the data and images for the web page to return its contents so your browser can display it. To do so, your computer will send an ASCII-text message called an HTTP request. Here's what a simple HTTP request for the data of an image might look like:


HTTP Request
GET /images/logo.png HTTP/1.1
Host: www.code.org
Client (e.g. browser)
Conducted over physical Internet using TCP/IP but HTTP protocol ignores these details
Server (e.g. website)
HTTP/1.1 200 OK
Content-Type: image/png
011101001…(binary of image)
HTTP Response

- **Method:** An HTTP request will begin with a method, which indicates what the client wants the server to do. The two most common methods are:



Method	Description
GET	Requests a specified web page or other data
POST	Submits some data for the server to accept or process

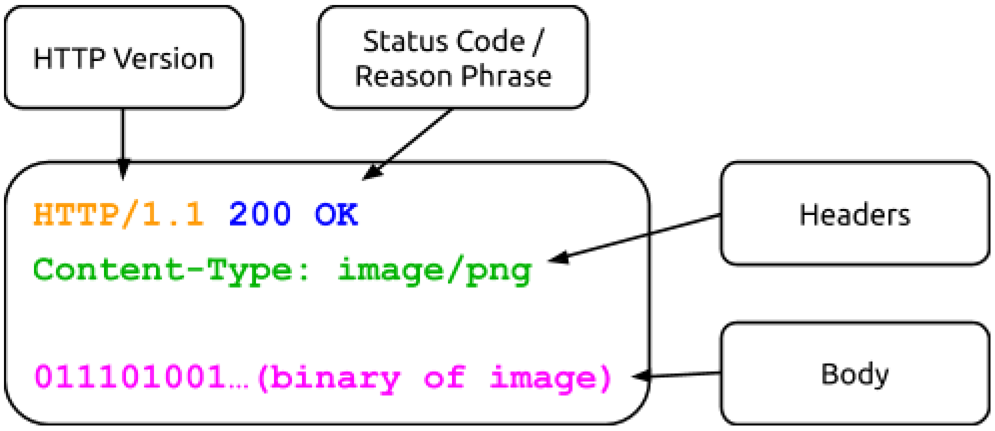
- **Resource:** The name of the resource you wish to access. In the example above, the request is for a .png image called "logo" located in the folder "images."

- **HTTP Version**: An indication of the version of HTTP being used; in this example it is HTTP 1.1.

- **Headers:** Additional information included to help the server interpret the request. In the above example, "Host" is included, but many more can be added by placing them on additional lines.

HTTP Response

When a server receives an HTTP request it will respond with a message of its own. Once again, the response will be sent entirely in ASCII-text and must be correctly formatted. Here's a sample HTTP Response:



An HTTP Response has multiple components:

- **HTTP Version:** An indication of the version of HTTP being used; in this example it is HTTP 1.1.

- **Status Code and Reason Phrase:** A number and phrase indicating how the server responded to the request. Some common Status Codes / Reason Phrases are:

Status Code / Reason Phrase	**Meaning**
200 Ok	Request was handled successfully
404 Not Found	Server could not find anything matching request
301 Moved Permanently	Requested data was permanently moved
302 Moved Temporarily	Requested data was temporarily moved
500 Internal Server Error	Error by the server prevented fulfilling request

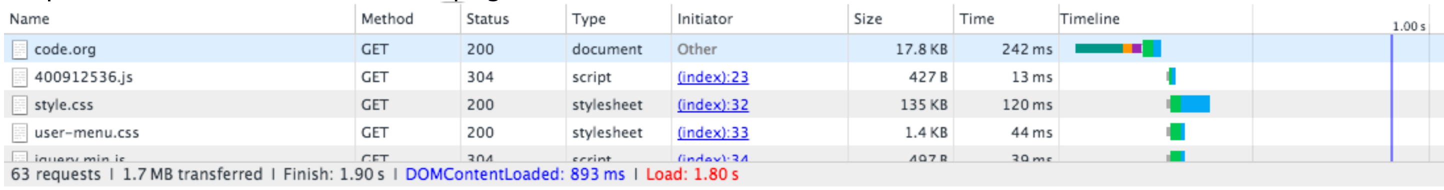
- **Headers:** Additional information about the response. The example above includes the header "Content-Type," indicating the type of content included in the response.

- **Body:** Following the headers, the response may include some data that was requested. In this example the actual binary representation of the image would be included in the response, so this response could be quite long.

Consider a "conversation" between you and a server. Indicate the steps your message takes on it is way to the server, then back to your computer. What information needs to be sent as part of your request, what information is returned?

☐ See HTTP in action

Most modern browsers include a set of Developer Tools designed to provide a detailed look into your browser’s activity. It is possible to monitor the traffic of HTTP requests and responses associated with a web page in real time.



Name	Method	Status	Type	Initiator	Size	Time	Timeline
code.org	GET	200	document	Other	17.8 KB	242 ms	
400912536.js	GET	304	script	(index):23	427 B	13 ms	
style.css	GET	200	stylesheet	(index):32	135 KB	120 ms	
user-menu.css	GET	200	stylesheet	(index):33	1.4 KB	44 ms	
jquery.min.js	GET	304	script	(index):34	497 B	39 ms	

63 requests | 1.7 MB transferred | Finish: 1.90 s | DOMContentLoaded: 893 ms | Load: 1.80 s

Use the links below to help you navigate to the Developer Tools of your browser. In Chrome, Internet Explorer and Firefox you’ll need to **open the “Network” tab.**
Chrome: https://developers.google.com/web/tools/chrome-devtools/
Internet Explorer: https://msdn.microsoft.com/library/bg182326(v=vs.85)
Firefox: https://developer.mozilla.org/en-US/docs/Tools/Network_Monitor
Safari: https://developer.apple.com/safari/tools/ (look at the “Network Requests” in the “Timelines” tab.)

Seeing HTTP in Action
You will use your browser’s developer tools to discover what kind of HTTP traffic is associated with visiting different types of websites. You and your partner are going to look at least 5 different types of websites:

1. http://example.com -- a very simple web page. Use this first to investigate developer tools.
2. **A “static” website** like: Wikipedia
3. **A news website** like: ESPN.com, BuzzFeed, the New York Times, etc.
4. **A streaming site** like: YouTube, or Spotify
5. **A site that accepts user input** like: twitter, facebook, email, google docs.

For each type of website below, follow these steps:

1. Monitor the HTTP traffic generated by loading the page.
2. Once the page has loaded, poke around with the developer tools and explore the different data coming in. What protocols can you see?
3. Interact with the website by clicking links or using other functionality on the site, noting how this affects the HTTP traffic.
4. For the five website you explore indicate the
    - Total amount of data received
    - Number of HTTP requests actually generated by loading one page
    - Total time to load the page.
    - Types of data received through HTTP (it’s more than just HTML)

Website 1:
Total amount of data received Number of HTTP requests actually generated by loading one page Total time to load the page. Types of data received through HTTP (it’s more than just HTML)
Website 2:
Total amount of data received Number of HTTP requests actually generated by loading one page Total time to load the page. Types of data received through HTTP (it’s more than just HTML)

Website 3:
Total amount of data received Number of HTTP requests actually generated by loading one page Total time to load the page. Types of data received through HTTP (it’s more than just HTML)
Website 4:
Total amount of data received Number of HTTP requests actually generated by loading one page Total time to load the page. Types of data received through HTTP (it’s more than just HTML)
Website 5:
Total amount of data received Number of HTTP requests actually generated by loading one page Total time to load the page. Types of data received through HTTP (it’s more than just HTML)

☐ Define key vocabulary

Write definitions for the following

Abstraction
Server
Client
HTTP
URL
Method
Header

☐ Receive Credit for the group portion of this lab

STOP

- Indicate the names of all group members.
- Have Ms. Pluska check your HTTP and Abstraction lab
- Submit your lab to the needs to be graded folder to receive credit for the group portion of this lab.
- Do not submit your lab until you have Ms. Pluska's (or her designated TA's) signature

____