HTTP

Scavenger Hunt

1. What is a **web resource**?

2. What are **4 Common Verbs** or **methods** used in the HTTP protocol?

3. What are HTTP headers? What is the **Accepts** header? What about **Content-type**? **User-agent**?

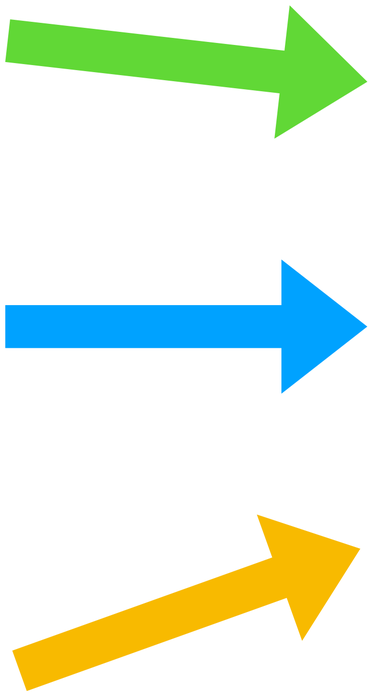
4. What is the meaning of 200-level **HTTP response codes**? Find 2 examples.

5. What is the meaning of 300-level **HTTP response codes**? Find 2 examples.

6. What are the meanings of 400-level **HTTP response codes**?
   500-level? Find 1 example in each range.

An HTTP Request Has

- Request Line
  - Method (GET/POST/PUT/DELETE)
  - Path
  - HTTP Version (e.g. HTTP/1.1)
- Headers
  - Key-Value string pairs delimited by “:”s and separated by new lines
- Body
  - If the request is giving content to the server (such as a form submission, application “post” or “save”)



```
POST /tweet HTTP/1.1

Host: api.twitter.com
Content-Type: application/json
Accept: application/json
Cookie: <CRYPTO_IDENTIFIER>

{“message”:“Hello, World”}
```

HTTP Server Receives the Request

- A reasonable mental model for HTTP is that of a (remote) function call
  - The pieces of an HTTP request are parameters to a server-side function
  - Very hand-waving psuedo-code:

```
let serverApp = (method, path, headers, body) => {
    // … server-side application logic …
    return new HTTPResponse(/* TBD */);
};
```

- Each piece of information in a request should be thought of as a

An HTTP Response Has

- Status Line
  - HTTP Version
  - Status Code (e.g. 200, 404, 500)
  - Reason Phrase (e.g. Ok, Not Found, Internal Server Error)
- Headers
  - Just like a request, key-value pairs delimited by ‘:’s and separated by new lines
- Response Body
  - Optional, but more common than in the client. For example, when a web page is requested its HTML comprises the response body.

HTTP Server Receives the Request

- Very hand-waving psuedo-code:

```
let serverApp = (method, path, headers, body) => {

    // … server-side application logic …

    return new HTTPResponse(statusCode, headers, body);
};
```