Codergirl - JavaScript

Class 15

~~October 19, 2020~~

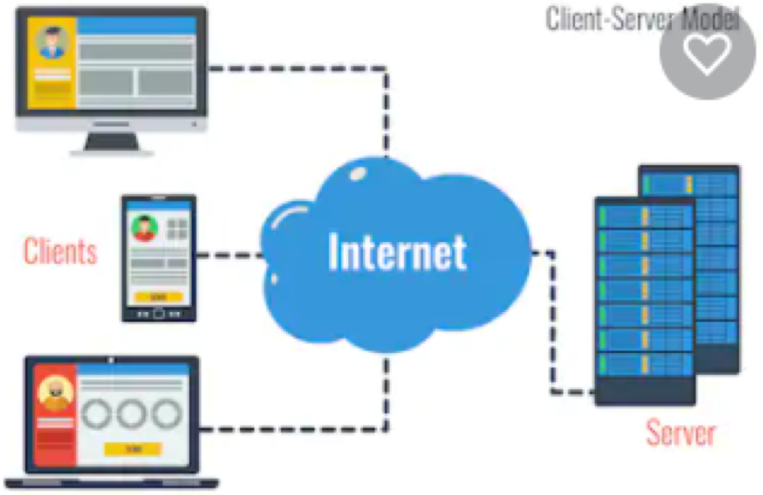
Agenda

- Studio coding activity – end at 6:10 pm
- Lecture – end at 6:30 pm
- Exercise – end at 7:00 pm

Studio – The DOM and Events

https://education.launchcode.org/intro-to-professional-web-dev/chapters/dom-and-events/studio.html

HTTP



- Client - Web browser
- Server - anywhere. (same computer)
- Server receives request and sends a response.
- Client displays meaningful info from response.
- Protocol is the way to send and receive info.

Protocols

Protocol	Full Name	Role
HTTP	Hypertext Transfer Protocol	High-level web communication for transferring files and information, including: • HTML, CSS, and JavaScript files • images and other media • form submissions
TCP/IP	Transmission Control Protocol / Internet Protocol	Low-level web communication for transferring small chunks of raw data known as *packets*
DNS	Domain Name Service	Translates human-friendly names into server addresses

Web address

scheme://host:port/path?query_string

http://www.launchcode.org/lc101?city=miami

HTTP

As long as the server is available, *every* request receives a single response.

Requests contain several types of data, including:

- The URL being requested.
- The type of action the client is asking the browser to take.
- Metadata about the request, such as the type of browser making the request and the type(s) of data the client can accept in return.
- Optionally, a request message.

Responses include:

- The status of the response, including success or failure reasons.
- Metadata about the response, including the size and data format of the response message.
- Optionally, a response message.

Requests

A generic HTTP request looks like this:

```
GET /blog/ HTTP/1.1
Host: www.launchcode.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:67.0) Gecko/20100101 Firefox/
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0

Request Body
```

The structure has these components:

1. **Request line:** The first line is the request line. It specifies the **request method**, path, and HTTP version being used.
2. **Request headers:** From line 2 until the first blank line, **request headers** are included as a series of key-value pairs, one per line.
3. **Blank line:** This signifies the end of the request headers.
4. **Request body (Optional):** Below the blank line, the request body takes up the remainder of the HTTP request.

HTTP Request methods.

GET – *retrieve* the resource.

Post – create new data.

- has body.

Headers

Header	Purpose	Example
`Host`	The domain name or IP address of the server the request should be sent to.	`www.launchcode.org`
`User-Agent`	Information about the client (usually a browser) making the request. The example is for a version of Firefox on a Mac.	`Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:67.0) Gecko/20100101 Firefox/67.0`
`Accept`	The types of data that the client is willing to accept in the response body.	`text/html,image/jpeg`
`Content-Type`	The type of data included in the request body. Usually only used for `POST` requests.	`application/json,application/xml`

Responses

```
HTTP/2.0 200 OK
Date: Wed, 22 May 2019 17:36:50 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 8050
Last-Modified: Wed, 22 May 2019 17:33:45 GMT

<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<!--Rest of HTML page -->
</html>
```

The structure has these components:

1. **Status line:** The first line of the response is the **response line**, which contains status information about the response including the **response code**. In this example, the response code is 200, which indicates the request was fulfilled successfully.
2. **Response headers:** Below the response line are the **response headers**. Similar to request headers, these are key-value pairs that contain metadata about the response.
3. **Blank line:** This signifies the end of the response headers.
4. **Response body (Optional):** Below the blank line, the request body takes up the remainder of the HTTP response. This is usually HTML, CSS, JavaScript, etc.

Common http response codes

Code	Description	Example
200	The requested resource exists and was successfully returned.	Visiting any existing web page on the Internet.
301	The requested resource has moved, and the client should look for it at the URL included in the `Location` header.	A site moves a page, but wants users with old links to be redirected to the page's new location.
404	The server received the request, but the requested resource does not exist on the server.	Requesting an image or HTML file that does not exist on the server.
500	The server experienced an error while fulfilling the request.	The server lost its database connection and cannot retrieve requested data.

Examples for HTTP

Forms

- Create a form – <form> tag
- <**input type**="text" **name**="username"/>
- **<label**>Username <**input type**="text" **name**="username"/></**label**>
- Value attribute

Form submission

- A submit button can be an input element with type=submit or a button element
- Method=‘POST’ or ‘GET’
- The action = where the request will be sent

Text inputs

Type	Syntax	Description	Demo
text	`<input type="text" name="username"/>`	A single line text field.	
textarea	`<textarea name="missionDescription"> </textarea>`	A larger, multi-line text box. Must have open and closing tags.	
password	`<input type="password" name="passCode"/>`	A text field that obscures the text typed by the user.	

Specialized Text inputs

Type	Syntax	Description	Demo
date	`<input type="date" name="flightDate"/>`	Browser validates the value is a valid date format. Some browsers provide a *date picker*.	mm/dd/yyyy
email	`<input type="email" name="emailAddress"/>`	Browser validates the value is a valid email address format.	
number	`<input type="number" name="fuelTemp"/>`	Browser validates the value is a valid number format.	

Checkbox and Radio Input

Type	Syntax	Description	Demo
checkbox	`<input type="checkbox" name="signUp"/>`	A small box for marking form option as *checked*.	sign up☐

Type	Syntax	Description	Demo
radio	`<input type="radio" name="crewReady" value="yes"/>`	A small circle that allows selecting *one* of multiple values. Used in groups of two or more.	yes○no○

Select Input

Type	Syntax	Description	Demo
select	`<select name="weather"> <option value="1">clear</option> <option value="2">cloudy</option> </select>`	A menu that allows selection of one option. Requires options to be in `<option>` tags.	clear

Validations

1. Add an event handler for the `window` *load* event
2. Within the window's load handler, add an event handler for the `form` *submit* eve
3. Retrieve input values that need to be validated from the DOM.
4. Within the form's submit handler, check the `input` values using conditional state
    a. If the values are valid, allow the form submission
    b. If the values are NOT valid, inform the user and STOP form submission

ach of these steps involves additional details, which we will now break down.

Examples and Exercises

Questions?

Studio time!