Intro to the Web

A web page is sent (copied) to your computer from the site

- Not like mainframes
- Not like apps

3 kinds of interactions

- Sending data and loading a new page
- Changing parts of the page in-place
- Sending/getting data w/o loading new

Trinity of the Web

- HTML is the structured content of the page
- CSS is the rules on how to look
    - based on the structure
- JS is the program adding in-page interactions
    - Runs ON YOUR COMPUTER
    - Only real option (for now?)

Backend is the program that gets/sends these pieces.

- On the SITE
- Can be any language
- Could be simple or complex

Intro to HTML

Is HTML a language?

- It is the "L" in the name

I mean, is it a programming language?

- "No", if you ask for vars+read/write+conditional
- "Yes", if you mean a syntax to instruct a computer
- But why are you asking?

Gatekeeping is not good, don't do it.

- programming is breaking down human-size problems to computer-size

HTML Example

HTML is the structured content of your page

- content
- structure
- NOT appearance

Browser provides a default appearance

MISTAKE to change structure based on default appearance

Browser and HTML

- Browser will guess for bad HTML
- MISTAKE to rely on this

Working, but not valid, webpage:

```
hello world
```

Try it in Chrome: File->Open

Basic page

```html
<!doctype html>
<html>
  <head>
    <title>Internet Cats</title>
  </head>
  <body>
    Hello World
  </body>
</html>
```

Rules of HTML Pages

The web is backwards compatible

- New browsers can open old pages
- Keeps everything working
- But how can we change behavior?

The "doctype" defines the set of rules

- browser will guess if not given
- "html" is HTML5, designed to evolve

```
<!doctype html>
```

HTML Tags

HTML is text that is "marked up" by **tags** that denote meaning.

- An **element** has opening and closing tags
    - may have content
- A tag is in *angle brackets*
    - The "p" tag: `<p>Hello World</p>`
- A separate closing tag has `/` after the opening angle bracket (`</p>`)

Closing HTML Tags

Some tags can "self-close"

- They have no contents
- opening and closing tag is the same tag
- the `/` is AFTER the tag name
    - `<div/>` the same as `<div></div>`
- A small number of tags don't require a closing
    - ...Because they hate us (not really)
    - `<br>`
    - Often people will close these anyway (`<br/>`)
        - technically wrong, but mostly harmless

Nesting Elements

- HTML elements **can** "nest"
- HTML elements **can't** "overlap"
- Elements can contain elements, text, and/or comments (`<!-- a comment -->`)

Valid: `<div><p></p></div>`

Not Valid: `<div><p></div></p>`

A handful of elements have additional rules

- Ex: Not valid to have `<p><div></div></p>` because paragraphs ("p" elements) can't have divisions.

Page of Cats

Internet is powered by cats

- Name of cat
- Picture of cats

We will worry about appearance later (CSS)

Start simple:

- Grumpy Cat
- Maru
- Nyan cat

Cat list, simple

```html
<!doctype html>
<html>
  <head>
    <title>Internet Cats</title>
  </head>
  <body>
    <ul>
      <li>Grumpy Cat</li>
      <li>Maru</li>
      <li>Nyan cat</li>
    </ul>
  </body>
</html>
```

What is UL?

Notice we have a `<ul>` element

- unordered list
- there's an order, it just isn't important
- want to guess what `<ol>` is?

The list is made up of individual **list items** (`<li>`)

Why not have many list items without `<ul>` element?

- how to separate two lists next to each other?

Semantic HTML

You don't want "just" HTML, you want "Semantic HTML"

Semantic means "related to meaning"

HTML where the structure is meaningful

- structure is not based on appearance
- structure is not ignored

More on this later, key lesson:

- Pick elements based on what they *mean*
    - not what they *look like*

HTML Rendering

The browser **renders** the HTML into a visible page

Browser provides default **styling**, even without CSS on the page

- **whitespace** is reduced to one space
    - we put in newlines and indents for humans
- In the case of invalid HTML browser will TRY HARD
    - and often succeed!
    - ...for now

HTML Attributes

Elements can have content and **attributes**

Attributes are additional data ABOUT the element

- url of related data
- how to behave
- labels, etc

attributes are put inside the opening tag

- space separated from tag name
- lowercase
- `name="value"` or just `name` format

Attribute examples

```html
<button disabled>Click Me</button>

<input type="checkbox">

<img src="https://examplecat.com/cat.png" alt="cat drawing">
```

Linking

The core of the web is actually LINKS

- originally a format to share and crosslink data like scientific papers

Before you can understand links, you have to understand URLs.

Uniform Resource Locator (URL)

A URL is an address (not just web, all of internet)

- what syntax to use (protocol)
- what port to use that on (port)
    - different protocols have default ports
- what computer to talk to (domain)
- what thing to request (path + file)

`http://northeastern.edu/wp-content/uploads/COE.jpg`

"Hey NEU server, I want `/wp-content/uploads/COE.jpg`"

Linking Pages

A link tells the browser to allow navigation to a different web page

```
<a href="http://neu.edu">Go to NEU</a>
```

"a" elements (anchor) have text content and an `href` attribute (hypertext reference) that says where to go when followed.

Let's create an "About Cats" page as a separate html file, and link to it from our Cat List page

Cat list, with link

```html
<!doctype html>
<html>
  <head>
    <title>Internet Cats</title>
  </head>
  <body>
    <a href="about.html">About Cats</a>
    <ul>
      <li>Grumpy Cat</li>
      <li>Maru</li>
      <li>Nyan cat</li>
    </ul>
  </body>
</html>
```

About Cats, with link

```
<!doctype html>
<html>
  <head>
    <title>About Cats</title>
  </head>
  <body>
    <a href="cats.html">Internet Cats</a>
    <p>
      This is a site about cats.
      What better way to master the web?
    </p>
  </body>
</html>
```

Not Fully Qualified

Why were those `href` so short?

We didn't use **fully qualified** urls. Anything we don't qualify as different is assumed to be the same as the current page.

- No protocol? Same protocol
- No domain? Same domain
- No path? Same path

Just listing the filename means it links to different files in the same directory

Relative vs Absolute

Common to omit protocol + domain

- Makes it easier to develop and move

File references can be **relative** or **absolute**

- Relative to current directory
- Absolute based on a **root** directory

The *root* is NOT the filesystem root, it is the webserver **document root**

Otherwise people could request any file on you computer

How to make absolute references

Absolute file references will always begin with `/`

- Sorry Windows users, the Internet is Unix-based

```html
<a href="/examplecat.png">See Cat</a>
<a href="/games/minecraft/data/guide.html">Punch Trees</a>
```

If it isn't absolute, it is relative

```html
<a href="about.html">About Us</a>
<a href="../dogs/why.html">Drool and barks</a>
```

Where to use URLs/references

Because life is not easy, different elements use references differently

- `a` tag uses `href`
- `img` tag uses `src`
- `link` tag uses `href` (e.g. to load CSS)
- `script` tag uses `src` (to load JS)

There's no real logic to the difference, you just have to remember/look up

```
<a href="https://examplecat.com/cat.png">A cat</a>
<img src="https://examplecat.com/cat.png">
```

So what is all of HTML?

Honestly, I don't remember it all. MDN is a good friend.

https://developer.mozilla.org/en-US/docs/Web/HTML/Element

Core elements:

- html
- head
- body

Common head elements

- title
- meta
- link
- style, script (more later)

Elements commonly in the body

- a
- b/strong, i/emphasis
- img
- p
- ol, ul, dl
- h1-h6
- div
- section, article, aside, header
- nav
- table elements
- various form elements (more later)

Table Elements

Back in the bad old days, tables were used to control the layout of web pages

DO NOT USE TABLES FOR LAYOUT

- Hard to understand
- Hard to change
- Semantically wrong
- a11y problems

Use tables for tables of data

ID attribute

Every element has an optional `id` attribute

If used, this should be an identifier that:

- is UNIQUE on the page
- says what the element is/contains (if by a human)
- is used by CSS/JS, but doesn't itself DO anything

Example ids:

```
<div id="root">...</div>
<form id="add_student">...</form>
<a href="/" id="homelink">Back to Home</a>
```

IDs are very common in examples for simplicity

Class attribute

`class` is a special attribute shared by all elements

This is a space-separated list of labels that:

- are NOT unique on the page
- should identify a category the element belongs to
- are used by CSS/JS, but don't DO anything

Example classes:

```
<img src="cat.png" class="cat demopic">
<a class="nav" href="about.html">About</a>
<p class="highlight">...</p>
```