- Current version is HTML 5.0

html Element

between <html> </html>

```
<!DOCTYPE html>
<htm>
    <head><title> My First html</title></
head>
<body>
Hello, World!<br>
<img src="http://..> //image
<audio src=""</audio> //audio
<video src="" </video>
<object data=""  type=""></object> //Multi
media
<iframe src="">


</body>
</html>
```

HTML (HyperText Markup Language)

Basic HTML

HTML is not case sensitive, but recommended the lower case

All element require opening and closing tag
Exception
<BR>
<IMG>

- Current version: HTML5
  - 1991: HTML(1)
    * Designed by Tim-Berner's Lee at CERNS
    * Based on SGML (Standard Generalized Markup Language)
  - 1995: HTML2.0, 1997: HTML3.2, 1998: HTML4.01
  - 2000: XHTML
  - 2014: HTML5
    * Standardization is both technical and political process
    * Dispute over future direction of HTML (XHTML vs HTML5)
    * All versions through HTML4.01 (and XHTML) were developed by W3C (World Wide Web Consortium)
    * HTML5 was developed by WHATWG (Web Hypertext Application Technology Working Group)

/Favicon.ico //logo of the website

Instead of
<b> use <strong>
<i> use <em> or <u>
<tt> use <code>

Insert whitespace => &nbsp;

Anything between < > is TAG

&lt; <
&gt; >
&amp; &

<!-- --> comments

- An HTML5 document starts with `<!DOCTYPE html>`
  - Triggers HTML5 standard mode in browsers
  - Earlier versions use different DOCTYPE
    * HTML4.01: `<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN""http://www.w3.org/TR/html4/strict.dtd">`
  - Remnants from SGML (Standard Generalized Markup Language)
- document = text + tags
  - Tags
    * special "markup symbols" enclosed in < ... >
  - Text
    * Any text not enclosed in < ... >
      - Q: How do we include < or > in text? What about &?
- Basic structure of an HTML document

```
<!DOCTYPE html>
<html>
<head><title>...</title></head>
<body>...</body>
</html>
```

HTML Elements

- HTML Element: a single HTML entity enclosed in an opening and a closing tag
  - Example: `<p>paragraph</p>`
  - An open tag `<x>` must be followed by a matching closing tag `</x>`, except "void elements" (= empty element) such as `<br>`, `<hr>`, `<img>`, ...
  - `<p>` is NOT a void element, but most browsers do not enforce closing it
- Tag names are case insensitive (lowercase is recommended)
- Common HTML tags
  - `<p>`, `<b>`, `<i>`, `<h1>`, `<ul>`, `<ol>`, `<li>`, ...
- Tags can have "attributes"
  - E.g., `<img src="...">`
  - Both single or double quotes can be used to enclose an attribute value
- Tags represent the document *structure*, not *formatting*
  - Most formatting tags like `<font>`, `<center>`, `<u>`, `<s>`, `<tt>` have been deprecated in HTML5
  - `<b>` and `<i>` tags still remain. Just too popular
  - Use CSS (Cascading Style Sheet) for formatting
    - Can be embedded using the `style` attribute like `style="font-family: courier"`
  - HTML5 has added many new "semantic elements"
    - `<header>`, `<footer>`, `<nav>`, `<article>`, `<section>`, ...
  - formatting vs structure
    - `<b>` -> `<strong>`
    - `<i>` -> `<em>`
    - `<tt>` -> `<code>`
    - `<s>` -> `<del>`
    - `<u>` -> `<ins>`

    * ...
  - Q: Why do we want to separate structure from style?

- Multiple white spaces and line breaks are displayed as a single white space
  - Q: How do we display multiple white spaces? line breaks?

- Comments appear in `<!-- ... -->`
  - Comment tags cannot be nested
- Q: how can we embed a link?
  - anchor tag: `<a href="url">...</a>`
    * note: `<link rel="relationship"href="url">` does not generate a clickable link
      ▸ e.g., `<link rel="stylesheet"href="style.css">` is for stylesheet
  - Fragment identifier: string behind # symbol in URL
    * points to the *HTML element* with the given id
    * http://a.com/a.html#g3 points to the element with `id="g5"` attribute on a.html
- Q: how can we embed a multimedia object? Image? Audio? Video? Flash?
  - Image: `<img ...>` tag
  - Audio: `<audio src="voice.mp3"type="audio/mpeg"controls>` tag
  - Video: `<video src="video.mp4"type="video/mp4"controls>` tag
    * `<audio>` and `<video>` tags were introduced in HTML5
    * Controls attribute add control buttons like play, pause, volume, etc
  - General object: `<object data="url"type="content-type">` tag

```
<object data="http://www.app.com/app.swf" type="
   application/x-shockwave-flash"></object>
```

    * `<embed src=url type=content-type>` in also very common (non-standard) due to browser compatibility issues
- Q: Can we include another HTML page in the page?
  - `<iframe>`
    * Example: `<iframe src="b.html"width="200"height="300">`

  - `<iframe>` is frequently used to get around the third-party-cookie restriction
    * more discussion later
- Q: where does the small icon next to title come?
  - favicon (favorite icon): small icon displayed next to title
  - default favicon path: `/favicon.ico`
  - nondefault favicon location can be specified through `<link rel="icon"href="path/to/favicon.png">`
    * recommended to be 196x196 pixels in size
- What's new in HTML5
  - New tags:
    * `<audio>`, `<video>`, `<header>`, `<email>`, ...
  - Clearly defined logic to translate "ill-defined" documents and handle errors
    * more consistent behavior among multiple browsers
  - Programmable Javascript API
    * Canvas element for 2D drawing
    * Web Storage for local data storage
    * Offline Web Application for offline app support
    * Document Editing and Drag-and-Drop
    * ...
- XHTML
  - Mostly the same as HTML, but much stricter formatting rules
    * tags and attributes MUST be lower case, not upper case.
    * ALL tags MUST have matching end tags. No empty elements. (e.g., `<br></br>`)
    * always use quotes around attribute values
  - Failed to take off because it was just too strict without much benefit to end users
- HTML validator
  - Online validators exist to check the standard compliance of a document
    * https://validator.w3.org/

User Input and HTML Forms

- Many Web sites want to generate contents dynamically based on "user input"
  - e.g., search by keywords, facebook status update, ...

  - Q: how can a server collect input from users?
- HTML form: an intuitive interface to get user's input
  - Example: Google search box http://oak.cs.ucla.edu/classes/cs144/examples/form.html

```
<form action="http://www.google.com/search" method="GET">
    <input name="q" type="text">`<input type="submit">
</form>
```

  - Interaction

```
        -- initial request -->
client  <- form page ---------   server
        -- request w input -->
        <-- result -----------
```

- `<form>`
  - action: the destination URL where input is sent (default: .)
  - method: HTTP method to use (default: GET)
- `<input>`
  - Show input type examples: http://oak.cs.ucla.edu/classes/cs144/examples/input-types.html
  - basic input types
    * simple box: text, password, . . .
      - Example: `<input type="text"name="username"value="initial value">`
      - `placeholder="hints"` attribute: hints is displayed if the field is empty
    * selection: checkbox, radio, and `<select>` + `<option>` tags
      - `checked` attribute for default selection
    * button: submit, button
      - `submit` submits the form while `button` does not
      - `value="text"`: button text
    * hidden: e.g., `<input type="hidden"name="email"value="a@b.com">`
      - Q: why do we need this?
  - Other popular input tags: `<textbox>`, `<button>`
    * `<textbox>` allows large text input area
    * `<button>` allows button text to be different from `value` attribute

  - HTML5 adds many more input types: date, time, email, color, number, . . .
- `<label>`
  - Example: `<label for="e1">Email:</label><input type="text"id="e1"name="email">`
  - `for="id"` attribute points to the `id` of the input
- User input encoding
  - every input field needs the `name` attribute
  - user inputs are encoded as a *"query string"*, which is `name=value` pairs separated by &
    * e.g., `first=John&last=Cho`
  - For GET
    * query string is attached to URL after ?, e.g., http://google.com/search?q=UCLA
    * General syntax of URL: `protocol://host/path?query_string#fragment_id`
  - For POST
    * query string is sent inside the request body
    * show example packet generated from http://oak.cs.ucla.edu/classes/cs144/examples/post.html

```
POST /search HTTP/1.1
...
Content-Type: application/x-www-form-urlencoded
Content-Length: 7
...
q=UCLA
```

- Sending a file to the server
  - `<input type="file">` allows the user to send a file to the server
  - `<form>` should use the following attributes:
    * `method="POST"`
    * `enctype="multipart/form-data"`
  - `multipart/form-data` MIME type
    * Makes it possible to upload files in a single request
    * Example at http://oak.cs.ucla.edu/classes/cs144/examples/multipart.html
    * Object "boundary separator" is specified as `boundary` attribute of Content-

Type header
  - E.g., `Content-Type: multipart/form-data; boundary=--EndOfFile`
* `Content-Disposition`: how to "dispose" each part.
  - possible values: `form-data`, `inline`, `attachment`

- Q: Exactly what happens at the server when they get HTTP requests?

References

- HTML5: https://www.w3.org/TR/html5/