## Chapter 2: Introduction to HTML Part 1

CS 80: Internet Programming

Instructor: Mark Edmonds

## HTML

- What does it stand for?
    - Hypertext markup language
- What is a markup language?
    - Not a traditional programming language
    - Specifies the structure and content of documents

## Editing HTML

- All you need is a text editor and a web browser
- Text editors
    - Brackets (recommended), Visual Studio Code, Sublime Text, Notepad++, TextEdit, Vim (!), etc
    - Anything will do, but don't use Microsoft Word (technically you can do this but it will make for a painful semester and you will almost certainly get points off)

## Editing HTML

- Web browsers
    - Chrome, firefox, IE, etc
    - For this course, firefox seems to be the most agreeable
        * There is one assignment with Ajax that only works with firefox

## HTML Concepts

- Documents are composed of HTML5 *tags*
- These tags outline this structure and content of a webpage
- Tag format:
    - `<tag>` begins the tag
    - `</tag>` closes the tag

**HTML Concepts**

- Most tags have a beginning and an end
  - Exceptions: `<meta>` (specifies document metadata, `<img>` tag (adds an image to the page), there are others we will come across
  - We call tags without an ending *void elements* because they do not markup text (text is not placed between a start and an end tag).

**HTML Concepts**

- Tags can be *nested* e.g. there is a tag inside of another tag before the outer tag's closing tag
- Attributes: content inside of a tag that specifies information about this particular use of the tag
- We will see an example in a moment
- Elements: the portions of a HTML document
  - The beginning to closing of a tag form an element
  - e.g. `<tag>` content `</tag>` is an element

**Example: `hello_world.html`**

```
 1  <!DOCTYPE html>
 2  <!-- document type declaration. required. must be on first line -->
 3
 4  <!-- starts the html document, the root of the document's structure.
        required. -->
 5  <html>
 6
 7  <!-- starts the head section of the document, provides info but usually
         not content. required. -->
 8  <head>
 9    <!-- metadata tag, here we specify the character encoding of the
          document. required. -->
10    <meta charset="utf-8">
11    <!-- specify the title of the document (what is displayed in the
          browser tab). required. -->
12    <title>Hello world!</title>
13  <!-- end of head section. required. -->
14  </head>
15
16  <!-- document body. the content goes here! required. -->
17  <body>
```

```
18    <!-- simple paragraph (the p-tag) -->
19    <p>Welcome to HTML5!</p>
20 <!-- end of body section. required. -->
21 </body>
22
23 <!-- ends the html document. required. -->
24 </html>
```

## HTML5 Validation

- You can validate HTML online!
- Go to https://validator.w3.org/#validate_by_upload to validate a file
- Go to https://validator.w3.org/#validate_by_input to validate input
- Upload `hello_world.html` example to validator

## Headings

- Heading elements designate a level of importance for a topic on a page
- Headings range from `<h1>` to `<h6>`
  - The lower the number, the greater importance
  - `<h1>` is the "most" important

## Example: `headings.html`

```
 1 <!DOCTYPE html>
 2 <html>
 3
 4 <head>
 5   <meta charset="utf-8">
 6   <title>Headings</title>
 7 </head>
 8
 9 <body>
10   <h1>Level 1 Heading</h1>
11   <h2>Level 2 heading</h2>
12   <h3>Level 3 heading</h3>
13   <h4>Level 4 heading</h4>
14   <h5>Level 5 heading</h5>
15   <h6>Level 6 heading</h6>
```

```
16    </body>
17
18    </html>
```

## Hyperlinks

- Provides a link to another HTML document
- Can be on this host or on a different host
  - What does this mean?
  - You can link internally to your own content or to a new host's webpage.
  - In either case, a new HTTP request is triggered

## Hyperlinks

- Links are facilitated using the `<a>` tag with a corresponding attribute of `href`
  - `a` stands for 'anchor'
  - `href` stands for 'hypertext reference'
- We can href other protocols (e.g. `https://`, `ftp://`, `mailto:`, `file:`, etc) or even javascript!
  - If it's javascript the script will execute when clicked
  - This will make more sense once we get to javascript, hold on!

## Example: `linking.html`

```
1    <!DOCTYPE html>
2    <html>
3
4    <head>
5      <meta charset="utf-8">
6      <title>Linking</title>
7    </head>
8
9    <body>
10     <h1>Welcome to CS 80 at SMC</h1>
11
12     <p>Hope you enjoy; make sure to <strong>turn in assignments on time</
         strong></p>
13
14     <p><a href="http://www.smc.edu/">SMC</a></p>
```

```
15    <p><a href="http://smconline.org/index.real?action=Login">eCompanion
        </a></p>
16    <p><a href="mailto:edmonds_mark@smc.edu">Email me</a></p>
17 </body>
18
19 </html>
```

## Images

- Added to HTML using the `<img>` tag
- `<img>` tag is one of the tags without an ending `</img>`
- The `src` attribute specifies where the image is located
    - The location can be a relative path (e.g. stored on the same computer as the html document)
    - The location can be a remote path (e.g. an image stored on a different host)

## Images

- Must use an `alt` attribute, it is allows for two important usages:
    - For those with poor or no eyesight to still understand the content on the page. A text-to-speech program can read the `alt` description to a visually impaired person.
    - If the image fails to load (maybe it's an external image), the `alt` can still describe what the image is
    - The `alt` description should be as brief as possible while still being descriptive

## Images

- Common attributes used: specifying height and width'
    - Height and width are both measured in pixels
    - If there are no height and width are specified, image will be rendered at it's own size (the original image size)
- Images can be nested inside of a link tag (`<a>`) to create an image that is also a hyperlink

## Example: `images.html`

```
1 <!DOCTYPE html>
2 <html>
3
```

```
 4  <head>
 5    <meta charset="utf-8">
 6    <title>Images</title>
 7  </head>
 8
 9  <body>
10    <h3>Every common image type is supported</h3>
11
12    <p><strong>Note: alt is a required attribute</strong></p>
13    <p>
14    <!-- relative path -->
15    <img src="./camel.png" alt="a camel" >
16    <!-- remote paths -->
17    <img src="https://cdn.meme.am/instances/500x/37834819.jpg" alt="The
          woes of programming" >
18    <img src="http://www.lakeshorebranding.com/wp-content/uploads
          /2011/12/web-design-chicago-438x275.png" alt="The woes of
          programming" >
19    </p>
20
21    <h3>We can control the width and height as well</h3>
22
23    <img src="https://media.giphy.com/media/eCqFYAVjjDksg/giphy.gif" alt=
          "Hacking in progress" height="200" width="350">
24
25    <h2>We can make our images links as well!</h2>
26
27    <a href="http://google.com">
28      <img src="https://cdn.vox-cdn.com/thumbor/YodYS9ma7P_8jcAplZSoIlw4v
            -c=/0x0:2012x1341/920x613/filters:focal(0x0:2012x1341)/cdn.vox-
            cdn.com/uploads/chorus_image/image/47070706/google2.0.0.jpg" alt
            ="The google logo."/>
29    </a>
30  </body>
31
32  </html>
```

## Special Characters

- HTML itself uses certain characters to represent the structure of the document
  - These symbols have special meaning within the document itself

        – What are some examples?
           * How are tags wrapped?
              · Around < and >

## Special Characters

- Clearly, we still want to be able to include these tags in an HTML document.
- Solution: We have to use a special convention to include these restricted characters
- These special conventions are called *character entity references*
  - The basic format: &`ref`;
    * Where `ref` is a reference to the character you wish to insert

## Special Characters

- E.g. &`lt`; inserts the less-than symbol (<)
- You can find a list of these symbols online
  - Do a quick google search go all of the symbols available
  - http://www.w3schools.com/html/html_entities.asp
  - https://www.w3.org/TR/REC-html40/sgml/entities.html
  - https://dev.w3.org/html5/html-author/charref

## Example: `char.html`

```
 1  <!DOCTYPE html>
 2  <html>
 3
 4  <head>
 5    <meta charset="utf-8">
 6    <title>Special characters and horizontal rules</title>
 7  </head>
 8
 9  <body>
10    <p>
11      <!-- use some special characters -->
12      We &amp;amp; can &lt; add &copy; symbols &trade; whereever &hellip;
           we &quot; want &gt; in &mdash; our &ndash; docment
13    </p>
14    <hr>
15    <p>
```

```
16      It's best practice to avoid the &lt;hr&gt; tag now because we can
          do styling with CSS, producing more professional looking and
          easier to maintain websites
17      We can also strike through <del>text</del>, subscript <sub>text</
          sub>, or superscript <sup>text</sup>.
18    </p>
19  </body>
20
21  </html>
```

## Lists

- Two types of lists:
    - Unordered list and ordered list
- Unordered list (bulleted list; like this list)
    - ul tag starts an unordered list (ul stands for unordered list)
    - Each list item is a nested li tag (li stands for list element)

## Lists

- Ordered list (numbered list)
    - ol tag starts an ordered list
    - each list item is a nested li tag
- We can also make lists inside of lists (nested lists)
    - Just like your word processor
    - Bullets of an unordered list will change based on the nesting level

## Example: `lists.html`

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5    <meta charset="utf-8">
6    <title>Lists</title>
7  </head>
8
9  <body>
```

```
10    <h2>We can make unordered and ordered lists!</h2>
11
12    <p>
13      This is an unordered list
14    </p>
15    <ul>
16      <li>this item has no order</li>
17      <li>neither does this one</li>
18      <li>this is not an exciting list</li>
19    </ul>
20
21    <p>
22      This is an ordered list with reasons why this course is good
23    </p>
24    <ol>
25      <li>Computer science is a great field</li>
26      <li>The course is fun!</li>
27      <li>The tests are take home</li>
28    </ol>
29
30    <p>
31      We can nest lists too
32    </p>
33    <ol>
34      <li>Computer science is a great field
35        <ul>
36          <li>It's fun</li>
37          <li>There are a lot of jobs</li>
38          <li>It makes you think</li>
39        </ul>
40      </li>
41      <li>The course is fun!
42        <ol>
43          <li>You get to explore an entire subfield in one semester
44            <ul>
45              <li>Can be the starting point for a career</li>
46            </ul>
47          </li>
48        </ol>
49      </li>
50      <li>The tests are take home
51        <ul>
52          <li>
```

```
53          A nest
54          <ul>
55            <li>
56              Another nest
57              <ul>
58                <li>Nests on nests</li>
59              </ul>
60            </li>
61          </ul>
62        </li>
63      </ul>
64    </ol>
65
66
67  </body>
68
69  </html>
```

## Line Breaks

- `<br>` tag - a line break

## Tables

- Similar to a textbook table or excel spreadsheet (except without math functions)
- Started with the `table` tag
- The nested `caption` tag gives the table a title and summarizes the table's content

## Tables

Table Bodies, Headers, and Footers

- `<thead>` and `<tfoot>` specify the header and footer of the table, respectively
  - Maybe you want different styling for the header and footer using CSS
  - Again, we'll cover that later

## Tables

Table Bodies, Headers, and Footers

- `<tbody>` specifies the main body portion of the table
- `<thead>` and `<tfoot>` have the same internal structure

## Tables

Table Rows

- `<tbody>`, `<thead>`, and `<tfoot>` is comprised of at least one `<tr>` element, which is a table row
  - Note that no table requires all three, but you should use them appropriately
- If your table has a header, put the header, etc.
- Makes table maintenance easier later

## Tables

Table Rows

- Each `<tr>` tag is comprised of `<th>` tags, which is a *header cell*. It is different from the normal cell!
  - It is different to allow easier styling using CSS
  - The table fills left to right along the columns
- Repeat this `<tr>` and `<th>` pattern for each row, and each column you want to specify
- `<tfoot>` can be below or above

## Tables

Table Rows

- `<tbody>` follows the same convention, but uses a `<td>` element instead of `<th>`
- `td` stands for 'table data'

## Tables

Table Sizing

- By default, each table column is only as wide as its largest cell.
- `rowspan` and `colspan`
- These attributes allows a cell to span multiple rows or columns
  - This is like merging rows/columns in a document

**Tables**

Table Sizing

- `rowspan` allows a single table cell to span the width of more than one cell or row
- `colspan` allows a single table cell to span the width of more than one cell or column
- These attributes can be applied in `<th>` and `<td>` elements

**Example: `tables.html`**

```
1   <!DOCTYPE html>
2   <html>
3   <!-- Fig. 2.12: table1.html -->
4   <!-- Creating a basic table. -->
5
6   <head>
7     <meta charset="utf-8">
8     <title>Tables</title>
9   </head>
10
11  <body>
12
13    <h2>Here's a simple table example</h2>
14
15    <h3>Notice the table scales to the widest cell's content (and not the
          caption's)</h3>
16
17    <!-- the <table> tag opens a table -->
18    <!-- Reminder: the border element should not be used. We are using it
          here before we learn CSS -->
19    <table border="1">
20      <!-- the <caption> tag summarizes the table's -->
21      <!-- contents (this helps visually impaired people) -->
22      <caption><strong>Table of Fruits (1st column) and Their Prices (2nd
          column)</strong></caption>
23      <!-- the <thead> section appears first in the table -->
24      <!-- it formats the table header area -->
25      <thead>
26        <tr>
27          <!-- <tr> inserts a table row -->
28          <th>Fruit</th>
29          <!-- insert a heading cell -->
```

```
30        <th>Price</th>
31      </tr>
32    </thead>
33    <!-- the <tfoot> section appears last in the table -->
34    <!-- it formats the table footer -->
35    <tfoot>
36      <tr>
37        <th>Total</th>
38        <th>$3.75</th>
39      </tr>
40    </tfoot>
41    <!-- within the <tbody> -->
42    <tbody>
43      <tr>
44        <td>Apple</td>
45        <!-- insert a data cell -->
46        <td>$0.25</td>
47      </tr>
48      <tr>
49        <td>Orange</td>
50        <td>$0.50</td>
51      </tr>
52      <tr>
53        <td>Banana</td>
54        <td>$1.00</td>
55      </tr>
56      <tr>
57        <td>Pineapple</td>
58        <td>$2.00</td>
59      </tr>
60    </tbody>
61  </table>
62
63  <!-- <br> is a line break -->
64  <br>
65
66  <h2>We can make more complicated tables using rowspan and colspan</h2
      >
67
68  <!-- Reminder: the border element should not be used. We are using it
        here before we learn CSS -->
69  <table border="1">
70    <caption>A more complex sample table</caption>
```

```
71      <thead>
72        <!-- rowspans and colspans merge the specified -->
73        <!-- number of cells vertically or horizontally -->
74        <tr>
75          <!-- merge two rows -->
76          <th rowspan="2">
77            <img src="camel.png" width="205" height="167" alt="Picture of
                  a camel">
78          </th>
79          <!-- merge four columns -->
80          <th colspan="4">
81            <strong>Camelid comparison</strong><br> Approximate as of
                  10/2011
82          </th>
83        </tr>
84        <tr>
85          <th># of humps</th>
86          <th>Indigenous region</th>
87          <th>Spits?</th>
88          <th>Produces wool?</th>
89        </tr>
90      </thead>
91      <tbody>
92        <tr>
93          <td>Camels (bactrian)</td>
94          <td>2</td>
95          <td>Africa/Asia</td>
96          <td>Yes</td>
97          <td>Yes</td>
98        </tr>
99        <tr>
100          <th>Llamas</th>
101          <td>1</td>
102          <td>Andes Mountains</td>
103          <td>Yes</td>
104          <td>Yes</td>
105        </tr>
106      </tbody>
107    </table>
108  </body>
109
110  </html>
```

**Forms**

- Mechanism for the user to send data to the server from the client
- The user types in the data, then hits 'submit' the submit triggers a
- HTTP request to the server to accept the form
    - The receiving end will be covered in Chapter 17
- Forms use the `<form>` tag

**Forms**

Important `<form>` attributes

- These setup the form and specify how and where to send data to the server
- `method` attribute can be two values:
    - `get`
    - `post`

**Forms**

Important `<form>` attributes

- `get`
    - Appends form data to url in name/value pairs
    - Length of URL is limited (~3000 characters limit)
    - Never use this to send sensitive data
    - Best suited for things like a query or other non-secure data (you can bookmark the url to effectively save the form submission)

**Forms**

Important `<form>` attributes

- `post`
    - Appends form data to HTTP request
    - Has no size limitations
    - Cannot be bookmarked (since it does not modify the URL)

**Forms**

Important `<form>` attributes

- `action`
  - Specifies where to send the form data (*e.g.* what site should process the form)
  - Must be a valid URL
  - The server **should** know how to respond to the form submission

**Forms**

Inputs

- Once we have the form setup, we can add inputs
- Inputs are added using the `<input>` tag

**Forms**

Important `<input>` attributes

- `type`
  - Controls the type of the input; a lot of options available
  - Options include: `text`, `button`, `color`, `password`, `radio`, `range`, `reset`, `submit`, etc. with more available online

**Forms**

Important `<input>` attributes

- `type`
  - `hidden` is a special type
    * It submits data to the server that is predetermined in the HTML page
    * The user cannot control this input
    * Might be used for sending information to another server to identify where the form is coming from

**Forms**

Important `<input>` attributes

- `type`
  - `text` has a couple of special attributes: `size` - specifies the size of the text box and `maxlength` which specifies the maximum length of the input

## Forms

Important `<input>` attributes

- `name`
  - Gives the input a name that can be referenced once the server receives the submission
  - This is part of the glue between the client and the server that processes the form

## Forms

Important `<input>` attributes

- `value`
  - Gives the input an initial value

## Forms

- Please lookup these elements in your book or online:
  - `textarea` - multiline text input, has rows and cols attributes
  - `password` - provides a password-protected field. this is only visually enforced (displays a * instead of the text), the password still should be encrypted when sent over HTTP
  - `color` - allows color input
  - `number` - allows user to input a number, similar to a text but for numbers

## Forms

- Please lookup these elements in your book or online:
  - `range` - allows user to pick between a range of values
  - `checkbox` - allows user to tick multiple options
  - `radio` - allows user to pick one option from a list
  - They all follow a similar pattern, but familiarize yourself!
- `<select>` tag presents a dropdown menu with a preselected list of options

**Example: `forms.html`**

```
 1  <!DOCTYPE html>
 2  <html>
 3
 4  <!-- Fig. 2.14: form.html -->
 5  <!-- Form with a text field and hidden fields. -->
 6
 7  <head>
 8    <meta charset="utf-8">
 9    <title>Feedback Form</title>
10  </head>
11
12  <body>
13    <h1>Feedback Form</h1>
14    <p>Please fill out this form to help us improve our site.</p>
15    <!-- this tag starts the the form, gives the -->
16    <!-- method of sending information and the -->
17    <!-- location of the form-processing script -->
18    <form method="post" action="http://www.deitel.com">
19      <!-- hidden inputs contain non-visual -->
20      <!-- information that will also be submitted -->
21      <input type="hidden" name="recipient" value="deitel@deitel.com">
22      <input type="hidden" name="subject" value="Feedback Form">
23      <input type="hidden" name="redirect" value="main.html">
24      <!-- <input type = "text"> inserts a text field -->
25      <p>
26        <label>Name:
27          <input name = "name" type = "text" size = "25" maxlength = "30"
              >
28        </label>
29      </p>
30
31      <p>
32        <label>Comments:<br>
33          <textarea name = "comments" rows = "4" cols = "36">Enter
              comments here.</textarea>
34        </label>
35      </p>
36      <p>
37        <!-- input types "submit" and "reset" insert -->
38        <!-- buttons for submitting and clearing the -->
```

```
39          <!-- form's contents, respectively -->
40          <input type="submit" value="Submit">
41          <input type="reset" value="Clear">
42       </p>
43     </form>
44
45   </body>
46
47   </html>
```

## Internal Linking

- Mechanism to jump between locations in a single document without reloading the HTML page
- Basic idea: we uniquely mark elements in the document using the id attribute, then we refer to the corresponding id in an anchor (a link, the a tag)
- The link tag can reference a specific tag in a different HTML document, even on a different host

## Example: `internal_linking.html`

```
1   <!DOCTYPE html>
2   <html>
3   <!-- Fig. 2.16: internal.html -->
4   <!-- Internal Linking -->
5   <head>
6     <meta charset="utf-8">
7     <title>Internal Linking</title>
8   </head>
9
10  <body>
11    <!-- id attribute creates an internal hyperlink destination -->
12    <h1 id="features">The Best Features of the Internet</h1>
13    <!-- an internal link's address is "#id" -->
14    <p><a href="#bugs">Go to <em>Favorite Bugs</em></a></p>
15    <ul>
16      <li>You can meet people from countries around the world.</li>
17      <li>You have access to new media as it becomes public:
18        <ul>
19          <li>New games</li>
20          <li>New applications
21            <ul>
```

```
22              <li>For Business</li>
23              <li>For Pleasure</li>
24            </ul>
25          </li>
26          <li>Around the clock news</li>
27          <li>Search Engines</li>
28          <li>Shopping</li>
29          <li>Programming
30            <ul>
31              <li>HTML5</li>
32              <li>Java</li>
33              <li>Dynamic HTML</li>
34              <li>Scripts</li>
35              <li>New languages</li>
36            </ul>
37          </li>
38        </ul>
39      </li>
40      <li>Links</li>
41      <li>Keeping in touch with old friends</li>
42      <li>It is the technology of the future!</li>
43    </ul>
44
45    <br><br><br><br><p>Skipping a whole bunch of space where you would
          put amazing web content</p><br><br><br><br>
46    <p>Skipping a whole bunch of space where you would put amazing web
          content</p><br><br><br><br>
47    <p>Skipping a whole bunch of space where you would put amazing web
          content</p><br><br><br><br>
48    <p>Skipping a whole bunch of space where you would put amazing web
          content</p><br><br><br><br>
49    <p>Skipping a whole bunch of space where you would put amazing web
          content</p><br><br><br><br>
50
51    <!-- id attribute creates an internal hyperlink destination -->
52    <h1 id="bugs">My 3 Favorite Bugs</h1>
53    <p>
54      <!-- internal hyperlink to features -->
55      <a href="#features">Go to <em>Favorite Features</em></a>
56    </p>
57    <ol>
58      <li>Fire Fly</li>
59      <li>Gal Ant</li>
```

```
60        <li>Roman Tic</li>
61     </ol>
62
63     <h1>We can even reference an id in a different HTML document on a
           different host</h1>
64     <p>
65       <a href="https://en.wikipedia.org/wiki/HTML#Attributes">https://en.
             wikipedia.org/wiki/HTML#Attributes</a>
66     </p>
67     <p>
68       <a href="#">Top</a>
69     </p>
70
71  </body>
72
73  </html>
```