Basic HTML

Written by Dave Pawlowski, November 29, 2011

1 Introduction

Hyper Text Markup Language is not a programming language. HTML uses markup tags (basically keywords) to describe web pages. Web browsers are built to read HTML documents and display them as web pages. The browser does not display the markup tag, but uses the tags to interpret the content of the page.

Web design has grown to be an incredibly complex field. It is no longer the norm to have a basic, static, web page written solely in html. Almost every page that you visit now is a dynamic page that responds to each users experience differently. Many webpages use html as a way to interact with underlying applications. For example, even my simple web-page, `http://chuck.emich.edu`, is run by a series of controller applications that most of the common tasks that a website would be asked to perform, such as clicking on a link.

Still, HTML is still used at least in some part in most websites on the internet. Thus having some knowledge of how to make a website is incredibly useful, even for a physicist. For one, you may find it necessary to have a professional website after you leave EMU. Additionally, It is becoming more and more commonplace for grand supported projects (especially those supported by federal grants) to require data to be made publicly available, usually via a website. This document will get you started. Just beware that web programming and web design can be a very complicated endeavor. There are entire university level programs devoted to getting people started in the field. That said, creating a basic document is relatively simple, as you will see here.

If you want to learn more about HTML and other web programming languages, `http://www.w3schools.com` is a good place to start. Once you learn some of the basics, one of the best things to do is find a site that you like, and view the source code. You can do this using any browser, though it varies, by clicking on something like the **view** menu, and then **source**. Google Chrome is setup with this in mind, and you can actually interact with the source code using their developer tools (**view** → **Developer** → **view source**).

2 Starting an HTML document

Any computer that is hooked up to the internet that has an ip address can host a website. For example, on chuck, everyone in class can host a website by creating a file called index.html in ˜/Sites. Open up index.html in emacs and well start adding some text and playing with a few tags.

First, HTML tags are used by web browsers to determine the layout, style, and print the contents of a html document. Tags are simply keywords surrounded by angle brackets <>. Every HTML document contains a few basic tags:

```
<html>
<body>

<h1> This is a heading </h1>

<p> This is a paragraph.
Note that a browser will ignore whitespace, for
the most part.</p>

</body>
</html>
```

Save the text file, and navigate your browser to chuck.emich.edu/ username, and you should see your website.

The `<html>`and `<body>`tags should be included in every document. The current HTML standards, adopted by the W3 consortium, specify that every tag should be closed, i.e. `<html>`must have a corresponding `</html>`. Most browsers don't strictly enforce this, but you will get different behaviors in different browsers if you don't close your tags.

In this simple example, you've also used one of the heading tags `<h1>`. Increasing the number following the h will cause the size of the heading to be a bit smaller.

```
<h1> This is a top level heading </h1>
<h2> This is a mid level heading </h2>
<h3> This is a lower level heading </h3>
<h4> Lower still... </h4>
```

Headings should not be used to increase the font size of regular text, or to make it bold. Headings are used by search engines to index the structure and content of your pages, so you don't want random text to show up to users that skim pages by its headings.

An important part of an html document is being able to create a link to another page:

```
<a href="http://chuck.emich.edu">Pawlowski's Homepage</a>
```

Here, the <a>tag is called an anchor, which performs several tasks. href is an attribute of the anchor tag.

Images can be included using the <img>tag:

```
 <img src="siena.jpg" width="100"/>
```

In this case, instead of using </img>to close the image, it is preferable to simply use \>after the element content.

3 Formatting

There are several different tags that enable text formatting:

```
<b> This text is bold! </b></br>
<big> This text is big! </big></br>
<em> Emphasized? </em></br>
<i> Italicized </i></br>
<small> itty-bitty text </small></br>
<strong> TOUGH TEXT!!! </strong></br>
and a <sub>sub</sub>script</br>
and a <sup>super</sup>script</br>
you may want to <ins>insert</ins> text</br>
or <del>delete</del> text</br>
this text looks like code... I guess: <code> for i in range(100)
</code></br>
```

Sometimes, you may want a browser to actually pay attention to white space. This is accomplished using the <pre>tag:

```
<pre>
Pre can be useful for coding:
for i in range(100):
   y[i] = y[i] + ode(y[i-1])
   t = t + i*h

print y

Since it maintains tabs, new lines, and such.
</pre>
```

4 A little more on linking

As you've seen, creating a link to a website is really simple using the <a>tag along with the **href** attribute. Another attribute that is useful is the **target** attribute.

```
<a href="http://www.emich.edu/" target="_blank">Eastern Michigan University</a>
```

will cause the link to open up in a new browser window or tab.

Another attribute that is useful is the **id** attribute. id allows you to name an anchor inside an HTML document, and then reference the anchor with in the document later on. For example, put the following at the top of your index.html:

```
<a name="upontop">The top of the page</a>
```

Then, at the bottom, add this link:

```
<a href="#upontop">Click here to go to the top!</a>
```

This can be used on external pages as well, assuming that the appropriate anchor exists already.

5 Tables

Tables are defined using the table tag, and is broken up into rows using the **tr** tag and the rows are divided into data cells, using **td**. A td can contain text, links, images, lists, forms and other tables.

```
<table border=’’solid’’>
<tr>
<th>Header 1</th>
<th>Header 2</th>
</tr>
<tr>
<td>row 1, cell 1</td>
<td>row 1, cell 2</td>
</tr>
<tr>
<td>row 2, cell 1</td>
<td>row 2, cell 2</td>
</tr>
</table>
```

Here, I’ve added the optional style attribute, border. The header row is also optional.

6 Lists

There are 3 main type of lists in HTML, unordered (bullets), ordered (numbers), definition (ed?):

```
<ul>
<li>Hockey</li>
<li>Football</li>
<li>Baseball</li>
</ul>
```

Will produce and unordered list. Swapping the ul with ol or dl will produce ordered lists or definition lists respectively.

7 Styling

Nearly every tag and attribute that has been mentioned in this document is capable of being styled. In other words, you can set the font of the list elements, or the type of border on a table, of change the color of the font in a paragraph. Each html tag may have several style attributes that you can specify when using the tag. **You should not do this!** While it is still supported by browsers, styling shouldn’t be handled this way. Instead, all styling should be handled using Cascading Style Sheets, or CSS. Doing this enables separation of content and styling, so that sites are more uniform, and the code that creates them is easier to understand by those that did not create it. css styling is usually done in a separate **stylesheet** that is linked to your html document in the header. CSS is easy to lean, but is not something we'll cover here. When you get comfortable with html, you can visit www.w3schools.com and learn about css, which has an excellent tutorial.