cis3.5, spring 2009, lab I.3 / prof sklar.

1 Creating a simple HTML page

1.1 Overview

- For this assignment, you will create an HTML file in a *text editor*.
  - on a PC, this is **Notepad** (not Wordpad)
  - on a Mac, this is **TextEdit** (in plain text mode, not rtf)
  - on Linux, this is **pico** or **emacs** or **vi**...
- Type your name in the files as *comments*.
  Comments in HTML are lines that begin with the tag: `<!--` and end with `-->`.
  For example:

  ```
  <!-- this is prof sklar's html file. -->
  <html>
  <body>
  hello!
  </body>
  </html>
  ```

- **Save your work!!!**
  BEFORE YOU LEAVE THE LAB, make sure that you save your work (both HTML files and any images files you use) by storing them on a USB flash drive or by emailing them to yourself (ask me if you don't know how to do this).

1.2 Getting started

- Open the appropriate text editor for the type of computer you are using.
  In this file, you will write HTML code to do the things listed below.
- Start by creating the outline for the page.
  All HTML pages begin with a `<html>` tag and end with a `<\html>` tag.
  In between these tags, the page is divided into two parts: the *head* and the *body*. These tags also have "start" and "end" forms: `<head>` and `</head>`, and `<body>` and `</body>`.
  Things that go inside the head tags do not appear directly in the content of the page, but are used to control the page in various ways. The content of the page goes between the body tags.
  The page outline should look like this:

  ```
  <html>

  <head>
  </head>

  <body>
  </body>

  </html>
  ```

- Set the **title** of the web page by inserting title tags (`<title>...</title>`) inside the head tags. When you view the HTML page, the title will be displayed inside the browser window's top heading bar, not in the body of the browser's content.

```
<head>
<title>prof sklar's sample home page</title>
</head>
```

- Set the **background color** of the page to something other than the default (usually white) and the **text color** of the page to something other than the default (usually black).
  You can do this using the **body** tag and the **bgcolor** parameter, like this:

  ```
  <body bgcolor="yellow" text="blue">
  ```

  Note that you will be *modifying* the body tag that you put in above, not adding another one. There should be only one body tag in an HTML document.
- Display a heading that welcomes the user to your web page. This needs to go in between the body tags.

  ```
  <body bgcolor="yellow" text="blue">

  <h1>Welcome to Prof Sklar's home page!</h1>

  </body>
  ```

- Save the file and name it with your last name followed by **-home.html**. For example, my file is called: **sklar-home.html**

  Note that some text editors, particularly on Windows, want to put a **.txt** extension after the **.html** extension. Don't do it.
- Try opening the file with a browser (like Netscape or Safari or IE) and see what your page looks like. In the browser, select **File** - **Open File...** and then enter the name of your HTML file (e.g., **sklar-home.html**).

  If any of the elements above don't work, go back into the text editor, and edit the file to fix them.

  Then reload the file in your browser.

  Keep doing this until everything looks just perfect :-)

  If you need help, ask me!!!

1.3 Adding content

- Between the heading (`<h1>...</h1>`) and the end body tag (`</body>`), add some text.
  All the page content goes in between the body tags.

  ```
  <h1>Welcome to Prof Sklar's home page!</h1>

  I'm having so much fun learning HTML!
  ```

1.4 Adding a list

- Now add HTML tags to display an **unordered list** of five of your favorite movies. For example:

```
My fave flicks are:
<ul>
<li> Casablanca </li>
<li> Diva </li>
<li> Bringing Up Baby </li>
</ul>
```

- Be sure to make the page user-friendly by putting in explanatory text telling what the list is (e.g., "Here is a list of my five of my favorite movies:").

1.5 Adding links

- A *link*, or "hyperlink", in HTML provides a means for going from one page to another. The link has two components:
  — one is the URL of the page that is being linked,
  — and the other is the text (or image) that the user will click on in order to follow the link.
  Here, we'll just put text for the user to click on.
- The HTML tag for a link is: `<a href=...URL...> ...CLICK... </a>`
  You need to replace the `...URL...` with the location of the page you want to link and `...CLICK...` with the text you want the user to click on in order to go to the linked page.
  For example:

```
<a href="http://www.amazon.com">amazon</a>
```

- Add links to each of the five items in your list of favorite movies. These could be links to the movies on Amazon or information on IMDB (`http://www.imdb.com`) or a trailer (e.g., `http://www.apple.com/trailers`) or some other relevant link.
  For example:

```
<ul>
<li> <a href="http://www.tcm.com:80/tcmdb/title.jsp?stid=568">
     Bringing Up Baby</a> </li>
</ul>
```

- **Save** your changes. **Reload** the page in the browser and try clicking on your link.
  If it doesn't work, then go back and fix the errors.

1.6 Adding images

- The HTML tag to include an image in your document is different from the other tags we've learned so far because it is a single tag; it does not have a begin tag and and end tag. The image tag is: `<img src=...URL...>`
- Find an image that you like on the web and then find the URL of that image. Usually right-clicking on the image from the browser window will give you the option to **Copy Image Location**. Do that and then **paste** the image location into your text editor window.
  Then enclose the URL in the image tag. For example:

```
<img src="http://g-ecx.images-amazon.com/images/G/01/gno/images/general/navAmazonLogoFooter._V264586593_.gif">
```

- **Save** your changes. **Reload** the page in the browser and try clicking on your link.
  If it doesn't work, then go back and fix the errors.
- The image tag has one required parameter, `src`, which is set to the URL of the image you want to display. The image tag has several optional parameters. A useful parameter is called `width` which allows you to define the width of the image, in pixels. Using this parameter, you can have the browser resize the image to fit the width you specify.

```
<img src=
"http://g-ecx.images-amazon.com/images/G/01/gno/images/general/navAmazonLogoFooter._V264586593_.gif"
width=200>
```

- Find an image to go with each of the movies on your list. Add an image tag to each list item. Set the width of all the images to be the same size.
- Again, **save** your changes. **Reload** the page in the browser and try clicking on your link.
  If it doesn't work, then go back and fix the errors.

1.7 Adding a table

- Tables provide ways of lining up information on an HTML page. The content of a table is organized in rows and columns. Rows extend horizontally. Columns extend vertically. See the illustration, below.

	column 1	column 2	column 3
row 1	A	B	C
row 2	D	E	F
row 3	G	H	J

- Tables are created using a series of tags.
  Table start and end tags `<table>...</table>` surround the whole table.
  Table rows start and end with: `<tr>...</tr>`
  Table columns start and end with: `<td>...</td>`
  The column tags are placed in between row tags.
  Below is the code to create a table with three rows and three columns:

```
<table border=1>
<tr> <td> A </td> <td> B </td> <td> C </td> </tr>
<tr> <td> D </td> <td> E </td> <td> F </td> </tr>
<tr> <td> G </td> <td> H </td> <td> J </td> </tr>
</table>
```

- Note that the table tag shown above specifies one parameter, `border`. This is set to a numeric value like 0 for no border, 1 for a thin border, 2 for a thicker border, etc.
- Try entering the sample table above in your HTML file. Save and reload to make sure it works.
- Then modify the sample table to have two columns and five rows. Repeat the information from your list of movies in the table.
  The first column of each row should contain the title of the movie and whatever link you used above.
  The second column of each row should contain the image.

2 Creating a simple style file

Cascading Style Sheets or **CSS** files provide a way to control the look and feel of your web page that is more convenient, more flexible and more comprehensive than adding parameters to HTML tags (like `<td bgcolor="blue">`). Style sheets let you define a wide range of settings such as colors, font types, font sizes, boxes around regions on your web page and spacing between various types of elements on the page. Style sheets are especially useful because you can create one CSS file that defines the look and feel of multiple web pages.

2.1 Getting started

A CSS file is a separate file from an HTML file. You create the CSS file in a text editor.

First you need to add a tag to the HTML file to tell it that it should use the CSS file. There are several ways to do this. One common way is shown below, using the **link** tag.

```
<head>
<title>prof sklar's sample home page</title>
<link rel="stylesheet" href="sklar-style.css" type="text/css">
</head>
```

Note that the `<link...>` tag is contained inside the `<head>...</head>` tags.

Add the link tag (as above) to your HTML file.

Next (below) you will create a separate CSS file that will contain the style sheet **rules**. The name of the CSS file should end in **.css** and the full filename is the value of the `href` parameter in the `link` tag. Above, my CSS file is named: **sklar-style.css**.

2.2 Setting body properties

Open a new file in your text editor.

Inside the file, enter a **body** rule to set the properties of the body element in the HTML file. The example below sets the background and text colors.

Note that these are the same properties that we set earlier, by using the `<body>` tag and its parameters `bgcolor` and `text`. You can set more properties using a style sheet. Note that the property settings in the style sheet take precedence over any set directly in the HTML tags.

```
body {
  background: red;
  color: brown;
}
```

Note the use of punctuation:

- the open-curly-bracket { at the beginning of the rule,
- the close-curly-bracket } at the end of the rule,
- the colon **:** after the name of each property being set by the rule, and
- the semi-colon **;** after the value of each property.;

Enter the rule, above, using your text editor. Make sure that you specify different colors here than the ones in your HTML `<body>` tag, so that you can be sure that the CSS file is being picked up by the browser.

When you are ready to save the file, name it with your last name followed by **-style.css**. For example, my file is called: **sklar-style.css**

Remember, this file name should match the value of the `href` parameter in the `<link>` tag that you entered in the previous step.

After you save both files (HTML and CSS), reload the HTML page in your browser. You should see the new colors for the background and text that you specified in your CSS file.

2.3 Setting table properties

You can use a style sheet to set the properties of a table. For example, the syntax for changing the colors of all the table cells and text is:

```
table {
  background: magenta;
  color: pink;
}
```

Table properties that can be set include:

name	description	possible values
color	color of text in table cells	any valid color name or hex value
background	background color for table cells	any valid color name or hex value
font-family	font specification for text in table cells	examples: serif, sans-serif, monospace, courier, verdana
font-size	size of font in table cells	relative value (e.g., 200%) or absolute (e.g., 8pt) values

Look at the on-line references listed at the end of this document for a complete list of table properties (see CSS Full Specification).

Try setting the color properties of your table by adding a **table** rule to your CSS file. Place it in the file after the **body** rule.

Actually, it does not matter which order rules are placed in a CSS file. The important thing is to keep the rule together, starting with the name of the rule, followed by the open-curly-bracket, the list of property settings and then the close-curly-bracket.

2.4 Setting list properties

You can use a style sheet to set the properties of a list. For example, the syntax for changing the background color of the whole unordered list is:

```
ul {
  background: black;
}
```

Try adding a **ul** rule to your CSS file, as above. Save the file and reload your HTML file in the browser. See how it looks.

You can also set the properties of list items.

```
li {
  font-size: 16pt;
  border: solid;
  border-width: thin;
}
```

This example shows some new properties: **font-size**, **border** and **border-width**.

Try adding this **li** rule to your CSS file. Save the file and reload your HTML file in the browser. See how it looks.

2.5 Dividing up your web page

One of the most powerful aspects of style sheets is the ability to divide your web page into sections, or "divisions". You can then set the properties within each division, regardless of whether you have lists, tables or whatever inside the division.

Using divisions requires two things:
(1) Specifying the extent of the division in the HTML file using `<div class=...NAME...> ... </div>` tags.
(2) Specifying properties for the division in your CSS file.

Bring up your HTML file in the text editor. Try adding a division containing the text content that you wrote in step 1.3, above.

```
<div class=fun>
I'm having so much fun learning HTML!
</div>
```

Save the HTML file. Note that if you reload it now, you will not see any difference in how it displays.

Now bring your CSS file into the text editor. Add a rule that sets the properties inside your "fun" class, for example:

```
div.fun {
  font-size: 24pt;
  background: green;
}
```

Now save the CSS file and reload the HTML in the browser. Now you should see a difference!

A few more things about divisions:

- You can name the divisions whatever you want, but the names should not contain any spaces. Use the underscore character (_) if you want to have a multi-word name, e.g., my_fun_division.
- Notice that in the CSS file, the rule begins with **div.** followed by the name you specified as the value of the `class` parameter in the `<div>` tag.
- You can "nest" divisions, by defining new divisions inside other ones. For example:

  ```
  <div class=main>
    <div class=part_one>
    this is the first part of the page.
    </div>
    <div class=part_two>
    this is the second part of the page.
    </div>
  </div>
  ```

  In this example, the divisions `part_one` and `part_two` are *nested* inside the division `main`.

On-line references

HTML

- Introduction to HTML:
  - `http://www.w3.org/MarkUp/Guide/Overview.html`
  - `http://htmldog.com/guides/htmlbeginner/`
- Advanced HTML:
  - `http://www.w3.org/MarkUp/Guide/Advanced.html`
  - `http://htmldog.com/guides/htmlintermediate/`
  - `http://htmldog.com/guides/htmladvanced/`
- HTML reference:
  - `http://htmldog.com/reference/htmltags/`
- HTML Full Specification:
  - `http://www.w3.org/TR/html4/` (HTML version 4.01)

CSS

- Introduction to CSS:
  - `http://www.w3.org/Style/Examples/011/firstcss`
  - `http://www.w3.org/Style/LieBos2e/enter/`
  - `http://www.w3.org/MarkUp/Guide/Style.html`
  - `http://htmldog.com/guides/cssbeginner/`
- Advanced CSS:
  - `http://htmldog.com/guides/cssintermediate/`
  - `http://htmldog.com/guides/cssadvanced/`
  - `http://www.w3.org/Style/Examples/007/`
- CSS Reference:
  - `http://htmldog.com/reference/cssproperties/`
- CSS Full Specification:
  - `http://www.w3.org/TR/css21/` (CSS version 2.1)

HTML validator:

- `http://validator.w3.org/`

Mobile devices:

- `http://www.w3.org/TR/2008/REC-mobile-bp-20080729/`
  (Mobile Web Best Practices 1.0)
- `http://developer.apple.com/iphone/`
  (iphone sdk – free download plus tutorials, videos and documentation)
- `http://code.google.com/android/`
  (android sdk – free download, documentation, etc)