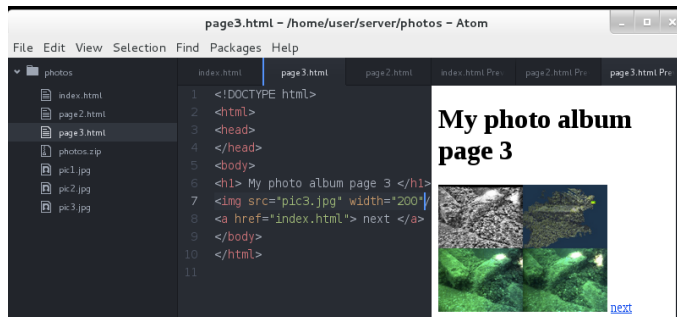
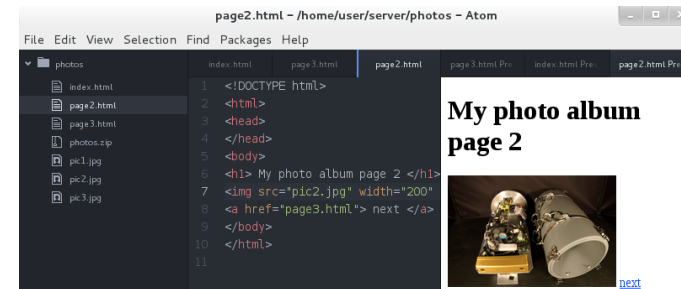
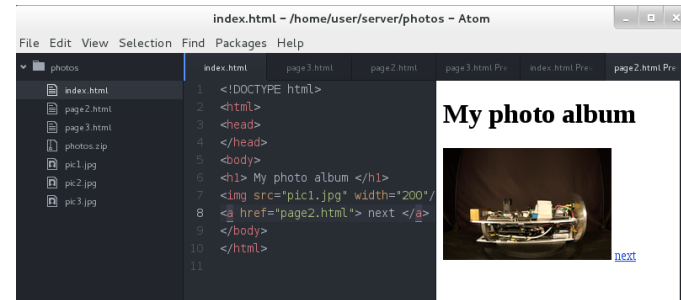


EECS 1012: Introduction to Computer Science

September 23, 2016



Download onto Android device

Adding some features

- Add a previous link
- Replace text links with images
- Structure into directories

Dealing with style

- When the web was younger, there was not a lot of attention paid to making pages render 'well' or 'in an intended style' on different hardware platforms.
 - That has changed...
- CSS (Cascading Style Sheets) are used to localize/compartmentalize/specify how the content of a particular web page should appear.
 - HTML -> content
 - CSS -> style

HTML5 & CSS

- In general, it is considered better style to separate style information (CSS) from content information (HTML) and to put both in separate files.
- Many advantages here, but one obvious one is that it enables style to be consistent over a web site.
- HTML files end in .html
- CSS files end in .css

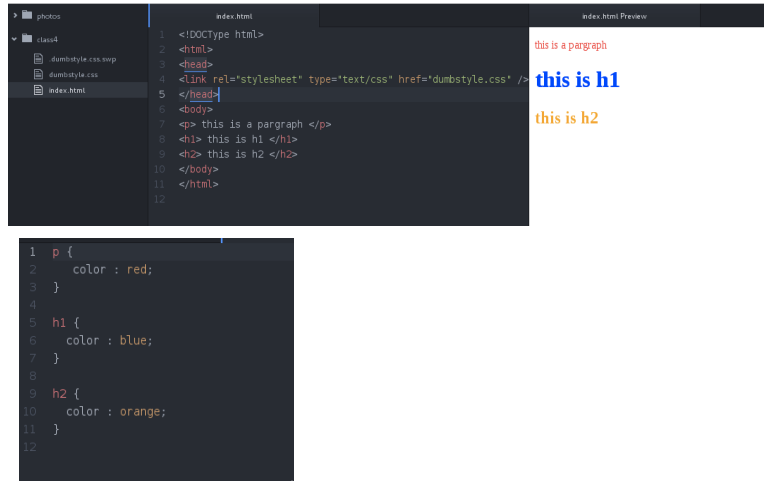
Link to external CSS from HTML

```
<head>  
<link rel="stylesheet" type="text/css" href="mystyle.css">  
</head>
```

CSS

- Cascading style sheet
- Goal is to separate content (text) from style (how it appears)
- It has a "C-like" syntax as opposed to HTML which has an XML (eXtensible Markup Language) syntax

So, a toy example



CSS file

- text file
- “white space does not count” (free format)
- c-style comments
- basic element is the selector

Selector

pattern { ← What to match against
property : value;
property : value;
...
}

p { ← match paragraphs
color : red;
}

Selector

pattern {
property : value; ← this property,
property : value; that value
...
}

p {
color : red; ← do things in red
}

Various selector possibilities

- name - that HTML element (element selector)
- #name - that ID (id selector)
- .name - that CLASS (class selector)

more options too, but lets start here

id

- <tag id="mytag"> </tag>
 - will match #mytag (only one)
- <tag class="myclass"> ... </tag>
 - will match .myclass (one or more)

Example



The screenshot shows a code editor with three panels. The left panel shows the HTML code for 'index.html', the middle panel shows the CSS code for 'dumbstyle.css', and the right panel shows a preview of the rendered HTML. The HTML code includes a DOCTYPE declaration, a link to the CSS file, and several elements with classes and IDs. The CSS code defines styles for these elements. The preview shows the rendered output with the text 'this is a paragraph', 'this is h1', 'this is h2', 'this is id1', and 'this is id2'.

```
index.html
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <link rel="stylesheet" type="text/css" href="dumbstyle.css" />
5 </head>
6 <body>
7 <p class="thisclass"> this is a paragraph </p>
8 <h1 class="thisclass"> this is h1 </h1>
9 <h2 class="thisclass"> this is h2 </h2>
10
11 <p id="id1"> this is id1 </p>
12
13 <p id="id2"> this is id2 </p>
14 </body>
15 </html>
16

dumbstyle.css
1 p {
2   color : red;
3 }
4
5 h1 {
6   color : blue;
7 }
8
9 h2 {
10  color : orange;
11 }
12
13 #id1 {
14   color : purple;
15 }
16
17 #id2 {
18   color : grey;
19 }
20
21 .thisclass {
22   color : green;
23 }
24

index.html Preview
this is a paragraph
this is h1
this is h2
this is id1
this is id2
```

Attributes

- Huge number, learn the ones you need
- Lets look at one or two here
- color : value;
 - value - some way of defining a colour

Color attribute

- there are about 140 colours defined by name (red, green, blue, orange, white, black, grey, midnight blue,)
- You can give by `rgb(r, g, b)` in range 0..255
- You can give by `rgba(r,g,b,a)` (`r,g,b` in range 0..255, `a` in range 0 (transparent)...1 (opaque))
- by hex
 - #00ff0
- Other choices too

Property background-color

```
1 p {
2   color : red;
3 }
4
5 h1 {
6   color : blue;
7 }
8
9 h2 {
10  color : orange;
11 }
12
13 #id1 {
14   color : purple;
15   background-color: violet;
16 }
17
18 #id2 {
19   color : grey;
20 }
21
22 .thisclass {
23   color : green;
24 }
25
```

this is a paragraph

this is h1

this is h2

this is id1

this is id2

So lets do something more meaningful

- Default `<body>` has some wasted space around it.
 - Lets get rid of it.
- Default `<body>` does not fill the entire screen
 - Lets make that happen

Units

- Can specify units of length (height) in absolute units
 - px (pixels ... sort of)
 - cm, mm, pt (1/72 of inch), others
- Can specify units in relative units
 - 10%, em (width of current font - often width of the 'm' character), rem (width of root - html font)

Android

```
26 body {  
27   width : 100%;  
28   height : 100%;  
29   padding : 0;  
30   outline : 0;  
31   border : 0;  
32   outline : 0;  
33   background-color : red;  
34 }
```

Use up all the space (although the red background is just to show it working)