Making our HTML Look Nicer
Cascading Style Sheets
Chapter 2

Supporting Material for “Using Google App Engine” - O’Reilly and Associates
www.appenginelearn.com
CSS Already?

- Back in the ancient days (i.e. before 2007) we got used to thinking that CSS was “advanced HTML” and that it was “difficult”

- With modern browsers, we should never bother with the old ways to change font, color, alignment, etc etc.

- CSS - Just Do It!
From Simple to Advanced CSS with no changes to the HTML
In the modern era of web design we represent content and meaning in HTML and formatting and layout in CSS.
Separation of concerns / specialization

Everyone needs to know some HTML and some CSS and some programming - but to be truly skilled at a professional level requires deep understanding and specialization.
HTML has evolved a *lot* over the years - as computers and networks have gotten faster.
CSS Syntax

- CSS Syntax is very different than HTML

- CSS is a set of “rules” which include a “selector” and one or more “properties” and “values” as well as some punctuation...

```html
body {
    font-family: arial, sans-serif;
}
```
Anatomy of a CSS Rule

selector - which part of the document does this rule apply to

property - which aspect of CSS are we changing

value - What are we setting the property to.

body {
  font-family: arial, sans-serif;
  font-size: 100%;
}

selector - which part of the document does this rule apply to

property - which aspect of CSS are we changing

value - What are we setting the property to.
Partial List of CSS Properties

- color
- background-color
- visibility (visible/hidden)
- font-family (arial, sans-serif)
- font-size
- font-style (italic, normal)
- font-weight (bold, normal)
- text-align
- vertical-align
- text-transform (lowercase, etc)
- text-decoration
- border-width
- border-style
- border-color
- margin
- border
- padding
- float (left, right, none)
- left / top
- position (static, relative, absolute)
- z-index

Applying CSS to our HTML

- **Inline** - right on an HTML tag - using the style= attribute
- An **embedded style sheet** in the <head> of the document
- As **external style sheet** in a separate file
Inline Styles

Google AppEngine is scalable.

Don't forget it is free (for small sites).

Google AppEngine is easy to learn.

We add a *style* attribute to a tag. We can have one or more properties and values. The styling applies for the scope of the tag.
The selector in the CSS rule indicates where in the displayed content is to be “styled” according to the properties.
External Style Sheet

- Preferred because it allows easy switching of style across many files.
- On larger teams, it also allows separation of concerns between the developer (HTML) and designer (CSS)
<head>
  <title>Learning the Google App Engine</title>
  <link type="text/css" rel="stylesheet" href="glike.css">
</head>
<body>
  ....

csev $ ls -l
total 32
  -rw-r--r--  1 csev  staff   44 Dec 19 06:06 glike.css
  -rw-r--r--  1 csev  staff  679 Dec 19 06:07 index.htm
  -rw-r--r--  1 csev  staff  883 Dec 19 05:59 sites.htm
  -rw-r--r--  1 csev  staff  679 Dec 19 05:59 topics.htm

csev $
First: Just work with the tags and fix fonts / colors
<head>
<title>Learning the Google App Engine</title>
<link type="text/css" rel="stylesheet" href="glike.css">
</head>
<body>

body {
    font-family: arial, sans-serif;
}

a {
    color: blue;
}

h1 a {
    text-decoration: none;
    color: black;
}
Select all “a” tags in the document.
<head>
<title>Learning the Google App Engine</title>
<link type="text/css" rel="stylesheet" href="glike.css">
</head>
<body>
<h1><a href="index.htm">AppEngineLearn</a></h1>
<ul>
<li><a href="sites.htm">Sites</a></li>
<li><a href="topics.htm">Topics</a></li>
</ul>
<h1>Google App Engine: About</h1>

Select tags where “a” is inside “h1”.
Marking Text for Styling

- Sometimes we cannot use an existing tag in the document for styling so we use the `span` tag.
- Sometimes we want to apply styling to *some* but all of the “p” tags using the `class=` attribute.
- Sometimes we want to break the document into “chunks” using the `id=` attribute.
Google AppEngine is scalable and don't forget that it is free for small sites and is easy to learn.

Span does not break word wrapping - it simply allows us to apply styling to a segment of text within a paragraph.
Making Some Tags Special

• We can use the class= to mark tags and then apply styling specifically to those tags.

• We style these sections by writing a rule which starts with a period followed by the name of the class

```html
<head>
  <style type="text/css">
    .abc {
      color: blue;
    }
  </style>
</head>
<body>
<p class="abc">First paragraph</p>
<p>Second Paragraph</p>
<p class="abc">Third paragraph</p>
</body>
```
Making Some Tags Special

First paragraph
Second Paragraph
Third paragraph

```
<html>
  <head>
    <style type="text/css">
      .abc {
        color: blue;
      }
    </style>
  </head>
  <body>
    <p class="abc">First paragraph</p>
    <p>Second Paragraph</p>
    <p class="abc">Third paragraph</p>
  </body>
</html>
```
Defining and Marking Blocks

- To mark out the scope of blocks, we use the `id=` attribute on a tag.
- We also have a new block tag called `<div>` that we often couple with the `id=` attribute on the `div` or other tag.
- While the `class=` attribute can be (re)used over and over on many tags in a document, a particular `id=` value can only be used once in a document.
- `id=` is used to mark a *particular* block in the document.
Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.
To apply styling rules to a block of text identified using id= we use a # character befor the id value.
<body>
  <div id="header">
    <h1><a href="index.htm">AppEngineLearn</a></h1>
    <ul>
      <li><a href="sites.htm">Sites</a></li>
      <li><a href="topics.htm">Topics</a></li>
    </ul>
  </div>
  <div id="content">
    <h2>Google App Engine: About</h2>
    <p>Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.</p>
  </div>
</body>
body {
    font-family: arial, sans-serif;
}

#header {
    background-color: #dde;
    border-top: 3px solid #36c;
}

a {
    color: blue;
}

#header h1 a {
    text-decoration: none;
    color: black;
}
Don’t Overuse Span/Div

• Don’t add unnecessary div tags when you already have a block tag to add a class to

• Don’t invent your own spans when something exists that will work

```html
<div class="important">
  <p>
    This is <span class="heavy">a</span> paragraph.
  </p>
</div>

<p class="important">
  This is <strong>a</strong> paragraph.
</p>
```
Color Names

- W3C has listed 16 color names that will validate with an HTML validator.
- The color names are: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow.

http://www.w3schools.com/html/html_colors.asp
Colors...

Three Numbers, Red, Green, and Blue - each from 00 - FF (Hexidecimal)

#ffffff = white
#000000 = black
#ff0000 = red
#00ff00 = green
#0000ff = blue

http://www.w3schools.com/css/css_colornames.asp
Web-Smart Colors

- Web-smart colors are 4096 colors which should easily display accurately on all systems with > 256 colors
- Web smart colors simply use any combination of 00, 11, 22, ... cc, dd, ee, ff
- #0055ee

http://www.morecrayons.com/palettes/webSmart
http://www.weborum.com/color/wheel.html
• Picking a combination of complimentary colors is also important - not all colors look good together

• This is a very artistic thing :)

http://www.colourlovers.com/
Fonts

- Default fonts are ugly and they have Serifs - which make them harder to read on a screen
- So the first thing I usually want to do is override the font in my document
- And I want to do this everywhere.
Fonts

Most Favourite | Least Favourite

body {
  font-family: "Trebuchet MS", Helvetica, Arial, sans-serif;
  font-size: x-large;
}

Fallback fonts: serif, sans-serif, monospace, cursive and fantasy.
Font Factors

font-size:
xx-small
x-small
small
medium
large
x-large
xx-large
14px

font-weight: bold or normal

font-style: normal or italic

none, underline, overline, or line-through
Styling for Links

Post-Click:

Hello there my name is Chuck.
Go ahead and click on here.

Downright Ugly!
Styling Links

link - before a visit
visited - after it has been visited
hover - when your mouse is over it but you have not clicked
active - you have clicked it and you have not yet see the new page

Browser default styling for links is downright ugly!
CSS CHEAT SHEET

SYNTAX

External Style Sheet
Syntax
<link rel="stylesheet" type="text/css" href="#" />
Internal Style
Syntax
<style type="text/css">
selector {property: value;}
</style>
Inline Style
Syntax
<tag style="property: value;"/>

GENERAL

Class
String preceded by a period
ID
String preceded by a hash mark
div
Format structure or block of text
span
Inline formatting
color
Foreground color
cursor
Appearance of the cursor
display
block, inline, list-item, none
overflow
How content overflowing its box is handled
visible
visible, hidden, auto

FONT

font-style
italic, normal
font-weight
bold, normal (lighter, bolder, integer [100-900])
font-family
Specific fonts(s) to be used

TEXT

letter-spacing
Space between letters
line-height
Vertical distance between baselines

BOX MODEL

Height
width
margin-top
margin-right
margin-bottom
margin-left
padding-top
padding-right
padding-bottom
padding-left

BORDER

border-width
Width of the border
border-style
dashed, dotted, double, groove, inset, outset, ridge, solid, none
border-color
Color of the border

POSITION

clear
Any floating elements around the element?
both, left, right, none
float
Floats to a specified side
left, right, none
left
The left position of an element
auto, length values (pt, in, cm, px)
top
The top position of an element
auto, length values (pt, in, cm, px)
position
static, relative, absolute
z-index
Element above or below overlapping elements?
auto, integer (higher numbers on top)

BACKGROUND

background-color
Background color
background-image
Background image
background-repeat

Web Developer Tools

- You pretty much need to use Firefox for serious website development
- Important plugins:
  - Web Developer - Chris Pedrick
  - FireBug - Joe Hewitt

http://addons.mozilla.org/
Using the Firefox Web Developer Plugin you can see the blocks which make up your page.
Web Developer

Plugin
Changing Page Layout in CSS
From Simple to Advanced CSS with no changes to the HTML
Two kinds of elements

- **Inline** - affects how text looks
  - strong, span
- **Block** - Containers that can be laid out
  - Paragraphs, etc
- CSS can change a tag from inline to block

```css
#navigation li {
  display: inline;
}
```
Inline Elements

• Flowed with other text

• span, em, strong, cite, a

• Inline tags can be nested as long as they match

• `<span class="important">`<cite>Stuff</cite>`</span>`

• Block can contain inline - but inline cannot contain block
Block Level Elements

• Starts on its own line - ends justification and starts a new block

• Can be a container for other elements

• h1 - h6, p, div, blockquote, ul, ol, form

• Blocks can contain other blocks

<div id=”content”>
<p>One</p>
<p>Two</p>
</div>
Now we will move things around, add background and borders, etc.
Think of a block as something to push around the screen.
Welcome to the site dedicated to learning the Google Application Engine.
We hope you find www.appenginelearn.com useful.
body {
    font-family: arial, sans-serif;
}

#header {
    background-color: #dde;
    border-top: 3px solid #36c;
}

a{
    color: blue;
}

#header h1 a {
    text-decoration: none;
    color: black;
}
Google App Engine: About

Welcome to the site dedicated to learning the Google Application Engine. We hope you find www.appenginelearn.com useful.

Next we will move these blocks around.
#header {
    background-color: #dde;
    border-top: 3px solid #36c;
    height: 100%;
    overflow:hidden;
}
#header li {
    font-size: 14px;
    display: inline;
}
#header {
  background-color: #dde;
  border-top: 3px solid #36c;
  height: 100%;
  overflow:hidden;
}

#header li {
  font-size: 14px;
  display: inline;
}

#header ul {
  list-style: none;
  float:right;
  vertical-align: middle;
}
#header {
    background-color: #dde;
    border-top: 3px solid #36c;
    height: 100%;
    overflow:hidden;
}
#header li {
    font-size: 14px;
    display: inline;
}
#header ul {
    list-style: none;
    float:right;
    vertical-align: middle;
}
#header h1 {
    font-size: 20px;
    float: left;
    vertical-align: middle;
}
Now lets do some “tweaking” to make it more visually pleasing...
CSS Box Model

- **height** and **width** properties size the block element
- **margin** properties define the space around the block element
- **border** properties define the borders around a block element
- **padding** properties define the space between the element border and the element content
- **background** properties allow you to control the background color of an element, set an image as the background, repeat a background image vertically or horizontally, and position an image on a page

http://reference.sitepoint.com/css/boxmodel
The Box Model

I am trapped in a glass case of emotion which is 100px high and 200px wide.

```html
<p class="trapped"> I am trapped in a glass case of emotion which is 100px high and 200px wide. </p>
```
Border, padding, and margin are additive.

```
.trapped {
    height: 50px;
    width: 50px;
}
.trapped2 {
    height: 50px;
    width: 50px;
    padding: 10px;
    border: 5px solid yellow;
}

<p class="trapped">
One</p>
<p class="trapped2">
Two</p>
```
```css
#header {
    background-color: #dde;
    border-top: 3px solid #36c;
    height: 100%;
    overflow: hidden;
    padding: 7px;
    margin-top: 5px;
}
#header h1 {
    font-size: 20px;
    float: left;
    vertical-align: middle;
    margin: 0;
    padding: 0 0 0 .3em;
}
#header li {
    font-size: 14px;
    display: inline;
    padding: .5em;
}
#header ul {
    list-style: none;
    text-align: right;
    float: right;
    vertical-align: middle;
    margin: 0;
    padding: 0;
}
```

`top, right, bottom, left`
Giving Navigation Clues...

App Engine: Sites
Here are some sites we hope you find useful:
- Python Learn
- Web Engine Learn
- Google App Engine Site

App Engine: Topics
- Python Basics
- Python Functions
- Python Python Objects
- Hello World
- The WebApp Framework
- Using Templates
<div id="header">
  <h1><a href="index.htm">AppEngineLearn</a></h1>
  <ul>
    <li><a href="sites.htm" class="selected">Sites</a></li>
    <li><a href="topics.htm">Topics</a></li>
  </ul>
</div>

#header li a.selected {
  color: black;
  text-decoration: none;
}

<div id="header">
  <h1><a href="index.htm">AppEngineLearn</a></h1>
  <ul>
    <li><a href="sites.htm">Sites</a></li>
    <li><a href="topics.htm" class="selected">Topics</a></li>
  </ul>
</div>
Tranforming the look and feel of a page using a CSS style sheet.

```
body {
    font-family: arial, san-serif;
}
a, a:link {
    color: #0000cc;
}
...
CSS Validation

- You can validate your CSS to make sure it has no syntax errors
- Browsers will generally quietly ignore bad CSS syntax
- [http://jigsaw.w3.org/css-validator](http://jigsaw.w3.org/css-validator)
- The validator can save you time and sanity
Zen Garden

- A social site where CSS designers show their “stuff”
- The HTML never changes - only the CSS
- Inspiration for us designers

http://www.csszengarden.com/
CSS Summary

• CSS Layout is its own art and science

• CSS Basics are well established and well supported in all modern browsers

• The box model is pretty straightforward - and allows nice design within the standards with reasonable effort levels.

• Site layout and markup is further evolving - mostly to make it increasingly possible to support desktop like experiences on the web.

• These innovations will naturally cause incompatibilities - which make things interesting and frustrating at times.