ART 263 Web Site Design
Basic Guide

Designing A Site

Every designer has his own design process, but it typically follows similar lines to those outlined below.

1. Establish and understand your client's needs, even if you are your client. Good communication between yourself and your client is key to producing a cost effective site that does not involve wasted time and money.
2. Look at the site content to get a feel for how you might start to develop a design concept.
3. Sketch out rough site layouts (wirferames) to establish spatial and scale relationships between content and an interface.
4. Create a flowchart that connects the different areas of the site and helps to establish an informational architecture.
5. Move to the computer and create several different prototypes (comps) for the site in Dreamweaver or Photoshop (usually three base designs are enough). At this time you will select:
    a. Your site palette – the colors used throughout the site. Limiting your palette to around 5-6 colors is usually sufficient.
    b. Your site typography – the font family and fonts used throughout the site.
    c. Images used for design purposes (that may or may not be part of the site's actual content).
6. Choose a design and continue to develop it in Dreamweaver.

Limitations

As you go through the design process above there are a few web-specific design issues you should consider:

1. Currently, the number of font faces available for use on the web is limited to system fonts shared between Macs and PC's (see handout *Working With Fonts*). CSS3 allows you to embed and link to fonts that have been converted to the .ttf and .eot formats, but they are not compatible with older browsers.
2. Logo's based on other fonts should be created in Photoshop and converted to images.
3. Web page images do not need to be a greater depth than 72dpi (screen resolution).
4. Blogs (unless your start from a site template), calendars, shopping carts, forms, guest books, etc. are more complex to create because they are based

on programming and scripting languages that have to be learnt in addition to HTML5. They may be based on languages such as PHP, PERL, and CGI.

5. Web sites very rarely display exactly the same way in all different browsers – especially so on devices and in very old browsers. When you design a web site you are aiming for a flexible design that can be viewed in many different browsers. The table below illustrates the current distribution of the most commonly used web browsers.

2014	IE11 (PC)	IE10 (PC)	IE9 (PC)	IE8 (PC)	IE7 (PC)	IE6 (PC)	Firefox	Chrome	Safari	Opera
Dec	3.6%+	1.1% -	1.6% -	1.3% -	0.2% -	0%	23.6% -	61.6% +	3.7%-	1.6%-

Web Page Layout

Page layout is controlled via a style sheet designed to control the way in which a page displays at various screen and browser window sizes. A style sheet is most often a linked .css (cascading style sheet) code file that determines the appearance of site content. It should be noted that though any web site could be designed using any of the layout types outlined below, choosing the *best* layout for your site design is critical to completing a web site in a timely manner and to accommodating specific audiences. You may not get it right the first time but, if styled and structured correctly, most site layouts can be quickly converted to another type.

Fixed (px)

Fixed layouts do not stretch or shrink as the browser window is resized. They are often used by web site designers who like precise positioning control, and in designs based on background images and other fixed-width content. Fixed width layouts should be limited to a maximum width of around 960 pixels to accommodate users on smaller resolution displays. One disadvantage to fixed-width layouts is that they do not make the best use of screen real estate on displays greater than 1024x768.

Fluid (%)

Fluid layouts (also know as *liquid*) expand or contract as the browser window is resized. As the browser window is resized page content shifts around to fill the window. One advantage of fluid layouts is that only at extremely small window sizes would scrollbars appear: Not having to scroll around a page to view content makes the page more accessible and user-friendly. Because of the inherent flexibility of fluid layouts, they are often the most challenging to design for and code. As a designer using this type of layout, your goal is to try to maintain the integrity of a design whilst leaving room for it to "shift" as the browser window resizes.

Elastic (em)

Elastic layouts expand or contract based on the text size a browser is set to use and on the base text size set in a style sheet. Elastic layouts use a unit of measurement

called an *em.* 1em = the font height. The layout width can fixed to a certain number of characters wide using ems. This means that there will always be the roughly same number of characters on a line but the actual line length (and therefore the width of the layout) increases or decreases based on font size changes.

Hybrid
Hybrid layouts use combinations of the above. For example, they may mix units of measurement; they may use fluid layouts but with fixed column widths (or vice-versa). They offer many of the advantages of the above layout types whilst avoiding many of their shortcomings. Because they employ mixed units of measurement they can often be a little tricky to code.

Some Important Web Design Concepts

Block-level vs. Inline
All HTML selectors (tags) are either block-level or inline. Each type can be redefined as the other through the CSS *display* property.

Block-level: have built-in line breaks before and after forcing an element onto its own line. *DIV* is an example of a block-level tag, as is P (for paragraph). NOTE: The first element in a web page BODY needs to be within a BLOCK-LEVEL tag in order for correct XHTML validation.

Inline: have no associated line breaks before or after and keep elements in line with each other. *SPAN* is an example of an inline tag, as is IMG (for image).

Class vs. ID
In CSS you can create two different user-definable selectors – class and ID.

Class: Classes can be associated with any HTML tag you choose and are used to define the appearance and/or position of an element. They can be used multiple times within the same web page, but can only appear once in a style sheet. You can recognize classes in a style sheet by the period (.) that appears before their names.

ID: ID's can be associated with any HTML tag you choose and are used to define the appearance and/or position of an element. They can only be used once within the same web page. You can recognize ID's in a style sheet by the pound symbol (#) that appears before their names. If you need to modify style sheet property values using Javascript then those properties need to be associated with an ID (because ID's are unique elements on a web page, whereas classes maybe repeatedly used).

File Naming
In HTML, files need to be named in a very particular way.

- All file names must end with an associated three-letter extension, except .html, which can be four.
- File names must not contain spaces.
- File names must not contain punctuation.
- File names are case-sensitive – they must replicate the way the file names appear as links on a web page.
- File names should be kept short. Though not essential, the shorter a file name the less typing involved and the less characters it takes up in a web page document.
- File names may use the underscore (_) and the dash (-) characters.
- File names may contain numbers.

Headings

Headings (as defined using the *h* tag) are numbered from 1 to 6. Contrary to what you may expect, a heading size 1 (*h1*) is the largest and a heading size 6 (*h6*) is the smallest; however, most designers do not use more than three or four headings sizes. Headings have built-in properties of a BOLD font weight and are block-level tags (see above) with top and bottom margins.

Images

The only permissible image file types for use on the WWW are:

GIF – Graphics Interchange Format (pronounced *jif*)
256 colors max. (8-bit)
May contain one transparent color
Best for graphics without gradients
Extension: .gif

JPEG – Joint Photographic Experts Group (pronounced *j-peg*)
Thousands of colors (16-bit)
Cannot have transparency
Can be saved at different qualities to reduce download times (optimization)
Best used for photographic images not containing transparency
Extension: .jpg

PNG – Portable Network Graphics (pronounced *ping*)
Can be 8-bit with a single transparent color or 24-bit with a transparent alpha channel giving various levels of opacity (not supported by older browsers and IE prior to version 7)
32-bit png files can be created in Adobe Fireworks
The greater the bit depth, the longer png files take to download
Extension: .png

SVG – Scalable Vector Graphics

SVG is the most recent image format for the web and is poorly implemented in several browsers. Image can be coded directly into a web page using HTML5's Canvas feature. Adobe Illustrator can save vector images in this format. The advantage of SVG images is that they can scale without deterioration and are smaller in file size, and therefore download more quickly. SVG files are not suited to photographic imagery.

Padding vs. Margins

Border: A line around a web page object. Borders set to a width of 0 are not visible. Borders separate padding and margins (see below).

Padding: The space between the content of a box and its border (even if the border is not visible). Think: *inside the box.*

Margin: The space around the outside of a box's border (even if the border is not visible) that separates different boxes on a page. Think: *outside the box.*

Mnemonics

Mnemonics are keys designed to help you remember things. There are two important mnemonics in CSS:

Box side order (applies to setting borders, padding, and margins)

Mnemonic: ***tr****ou****bl****e* – top, right, bottom, left

Anchor tag state order (pseudo link classes in your style sheets need to be listed in this order)

Mnemonic: ***l****o****v****e-****ha****te* – link, visited, hover, active

Web Page Structure

A web page is constructed from a series of TAGS. Tags are HTML formatting elements that define sections of a web page. The two different types of tag are INLINE and BLOCK. Inline elements have no line breaks associated with them whereas block level tags have a preceding and a following line break. Block level tags are the basis for page layout, inline tags are used to mark up content *contained by those blocks*.

HTML = defines where a web page begins and ends.
HEAD = contains the non-visible part of a web site – information that the browser needs to display a page correctly and information for search engines.
LINK = used to connect a style sheet to a web page.
META = used to convey hidden information about a web page and provide information for search engines.

BODY = all of the content that is a part of your page goes into the body (which is derived from the publishing term "Body Copy."
DIV = is used to divide a page into logical areas that can be freely positioned. DIVs are assigned IDs to give them a unique identity. Without an ID a DIV cannot be told where to go or how to appear. DIVs are block level tags and can be nested (one DIV placed within another DIV.) DIVs are often referred to as *containers, blocks, or boxes*.

Semantic Markup in HTML5

Semantic markup defines the meaning of the content located within that markup. This is not a new principle to web site design, but it is one that is emphasized in HTML5. Below is a table of the tags available for markup under HTML5. NOTE: The table is included for completeness – not all tags are currently available for use in all browsers and they will not all be covered or used in this course. The table is primarily useful for reference purposes only.

HTML5 Basic Tag Reference

Tag	Description
<!--...-->	Defines a HTML comment (not visible on a web page when viewed in a browser)
<!DOCTYPE html>	Defines the web page document type
<a>	Defines a hyperlink
<abbr>	Defines an abbreviation
<address>	Defines the web page contact person's email address or other contact information
<article>	Defines an area of a page in which all of the content is related
<aside>	Defines additional information relating to the main content
<b>	Defines stylistically offset text – keyword, product name, lead sentence or paragraph
<blockquote>	Defines a long quotation
<body>	Defines the body (visible content area) of a web page
 	Inserts a single line break, used for poems and addresses
<button>	Defines a push button (form element)
<canvas>	Defines graphics
<caption>	Defines a table caption
<cite>	Defines a citation, the title of a work
<col>	Defines attributes for table columns
<colgroup>	Defines groups of table columns
<dd>	Defines a definition description
<dl>	Defines a definition list
<dt>	Defines a definition term
	Defines emphasized text
<fieldset>	Defines a fieldset (form element)
<figure>	Defines a containing box for an illustration and caption
<figcaption>	Defines a caption to accompany a figure
<footer>	Defines a footer for a section or page
<form>	Defines a form (form element)

<h1> to <h6>	Defines heading a from size 1 to 6
<head>	Defines information about the document
<header>	Defines a header for a section or page
<hgroup>	Defines headings with subheadings
<hr />	Defines a horizontal rule used for a paragraph thematic break
<html lang="en">	Defines an html5 document
<i>	Defines text which indicates an idiom, technical term, ship names, terms from taxonomies, etc.
<iframe>	Defines an inline sub window (frame)
<img />	Defines an image
<input>	Defines an input field (form element)
<label>	Defines a label for a form control (form element)
<legend>	Defines a title in a fieldset (form element)
	Defines a list item
<link />	Defines a resource reference
<map>	Defines an image map
<nav>	Defines navigation links
	Defines an ordered list
<optgroup>	Defines an option group (form element)
<option>	Defines an option in a drop-down list (form element)
<p>	Defines a paragraph
<pre>	Defines preformatted text
<q>	Defines a short quotation
<section>	Defines part of an article or an area of a page
<small>	Defines small text used for disclaimers, copyright info, etc.
	Defines strong text, which means importance
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<table>	Defines a table
<tbody>	Defines a table body – located just within the table tag
<td>	Defines a table cell – located within the tr tag
<textarea>	Defines a text area (form element)
<tfoot>	Defines a table footer
<th>	Defines a table header
<title>	Defines the document title
<tr>	Defines a table row – located just within the tbody tag
	Defines an unordered list

Video and Audio

Under HTML5 video and audio is supported without the need for browser plugins. Different browsers support one if two formats: OGG or MP3 for audio and OGG/OGV or MP4 for video. As a result you must convert your media files to both formats to support the various browsers. The browser will only load the version it supports.

Attribute	Value	Description
autoplay	autoplay	Specifies that the audio should start playing as soon as it is ready
controls	controls	Specifies that playback controls should be displayed

`loop`	loop	Specifies that the audio should start over again, when it is finished.
`preload`	auto metadata none	Specifies whether or not the audio should be loaded when the page loads
`src`	*url*	Specifies the URL of the audio to play

Attribute	**Value**	**Description**
audio	muted	Specifies the default state of the the audio. Currently, only "muted" is allowed
autoplay	autoplay	If present, then the video will start playing as soon as it is ready
controls	controls	If present, controls will be displayed, such as a play button
height	*pixels*	Sets the height of the video player
loop	loop	If present, the video will start over again, every time it is finished
poster	*url*	Specifies the URL of an image representing the video
preload	auto metadata none	Specifies whether or not the video should be loaded when the page loads
src	*url*	The URL of the video to play
width	*pixels*	Sets the width of the video player