

CIS 4004: Web Based Information Technology Spring 2011

Basic Page Layouts – Part 1

Instructor : Dr. Mark Llewellyn
markl@cs.ucf.edu
HEC 236, 407-823-2790
<http://www.cs.ucf.edu/courses/cis4004/spr2011>

Department of Electrical Engineering and Computer Science
University of Central Florida



Basic Page Layout

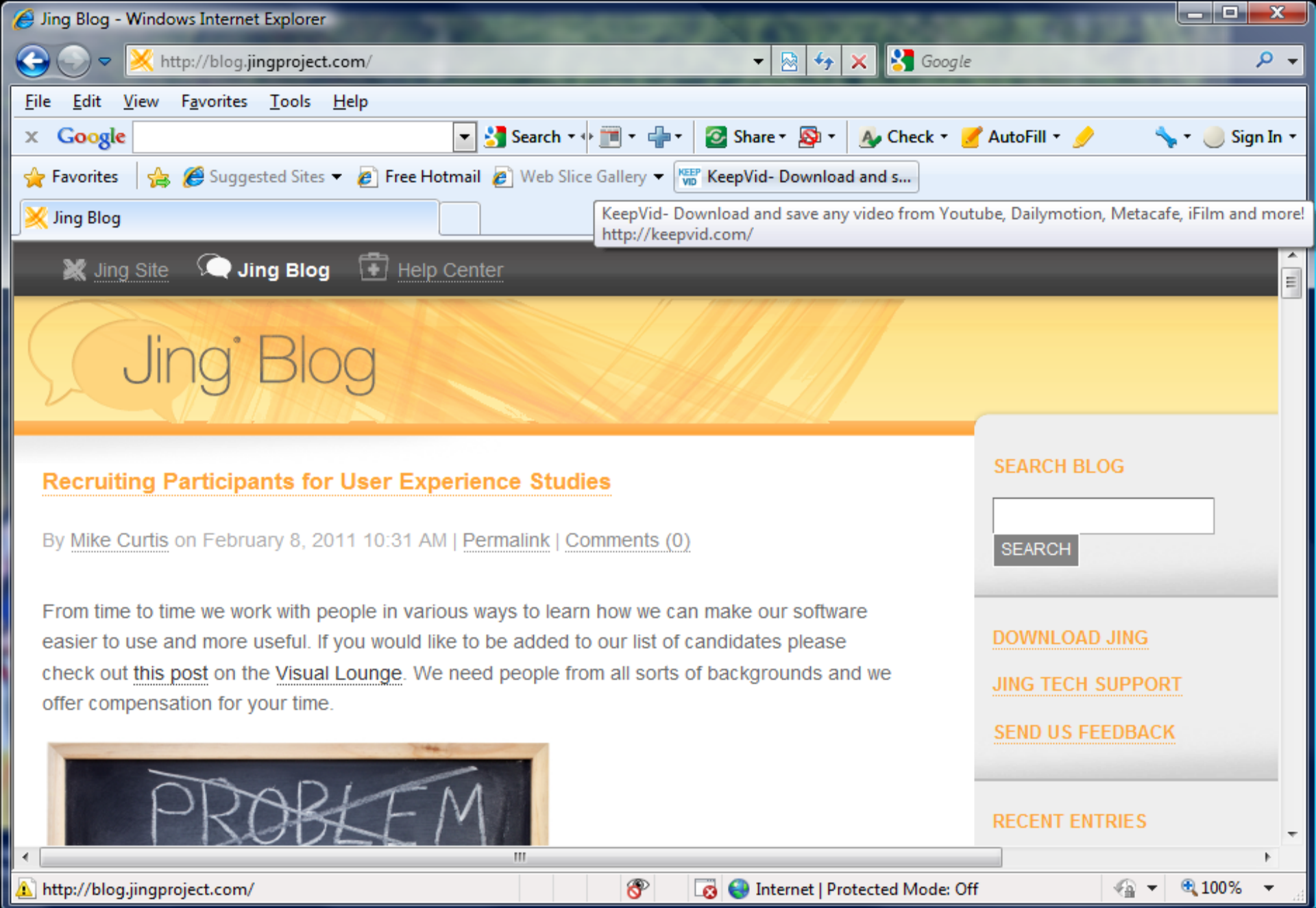
- If you examine the markup of most site designs, they are based on two or three column layouts, even though that fact is sometimes visually well disguised.
- In this section of notes, we'll examine the way these layouts are created using XHTML and CSS. We'll worry about adding more visual components to the layout later, right now we want to focus on the underlying structure of the layout.
- We'll examine both fixed-width and liquid layouts (where the layout expands automatically to fill the browser window).



Multi-column Layouts

- A basic use of columns is to organize a list of navigation links down the left or right side of the page next to the main content area.
- An example of this is the Jing page (Jing is a new screen recording technology currently under development). The Jing blog at <http://blog.jingproject.com> is an excellent example of a liquid two-column layout. Their page is shown on the next two slides.
- Visit this site and then adjust the width of your browser to see the “liquid” effect. The main content area changes its width as the browser changes size. The text automatically rewraps to the new line length as the width changes.





[Recruiting Participants for User Experience Studies](#)

By [Mike Curtis](#) on February 8, 2011 10:31 AM | [Permalink](#) | [Comments \(0\)](#)

From time to time we work with people in various ways to learn how we can make our software easier to use and more useful. If you would like to be added to our list of candidates please check out [this post](#) on the [Visual Lounge](#). We need people from all sorts of backgrounds and we offer compensation for your time.



SEARCH BLOG

- [DOWNLOAD JING](#)
- [JING TECH SUPPORT](#)
- [SEND US FEEDBACK](#)

RECENT ENTRIES



Jing Blog - Windows Internet Explorer

http://blog.jingproject.com/

File Edit View Favorites Tools Help

Google Search Share Sign In

Jing Blog


Jing Site Jing Blog Help Center

Jing Blog

Recruiting Participants for User Experience Studies

By [Mike Curtis](#) on February 8, 2011 10:31 AM | [Permalink](#) | [Comments \(0\)](#)

From time to time we work with people in various ways to learn how we can make our software easier to use and more useful. If you would like to be added to our list of candidates please check out [this post](#) on the [Visual Lounge](#). We need people from all sorts of backgrounds and we offer compensation for your time.



SEARCH BY

SEARCH

[DOWNLOAD](#)

[JING TECH](#)

[SEND US F](#)

RECENT EN

Done Internet | Protected Mode: Off 100%

Browser window smaller in size – note text wrapping compared to previous slide.





Browser window smaller in size – note text simply disappears



Multi-column Layouts

- Notice that when the browser is shrunk too much, that you reach a point where the layout will become no smaller and the right side of the browser window simply covers it up.
- We'll see how to implement both of these effects – liquid layout and set a minimum width – as we look at layouts in more detail.



Multi-column Layouts

- Perhaps the most common layout is three columns, typically with left navigation, content in the middle, and the right column commonly used for advertisements, links to other sites, headlines, etc.
- A good example of a three-column layout is Amazon.com. They put the navigation in the header so it can use two columns to tell you about the product and the third column to help you buy it.





Hello, Mark Llewellyn. We have [recommendations](#) for you. (Not Mark?)

[Halloween Deals on Costumes and More](#)

[Mark's Amazon.com](#) | [Today's Deals](#) | [Gifts & Wish Lists](#) | [Gift Cards](#)

[Your Account](#) | [Help](#)

Shop All Departments

Search Books

GO

Cart

Wish List

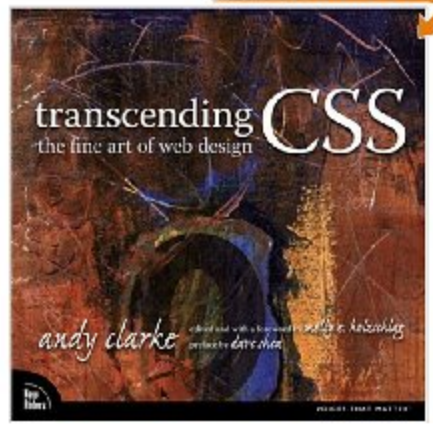
Books Advanced Search Browse Subjects New Releases Bestsellers The New York Times® Bestsellers Libros En Español Bargain Books Textbooks

Prime

Mark Llewellyn: This item is eligible for Amazon Prime. [Click here to turn on 1-Click](#) and make Prime even better for you. (With 1-Click enabled, you can always use the regular shopping cart as well.)

Member: Mark Llewellyn

Click to LOOK INSIDE!



Transcending CSS: The Fine Art of Web Design (Paperback)

by [Andy Clarke](#) (Author), [Molly E. Holzschlag](#) (Author), [Aaron Gustafson](#) (Technical Editor), [Mark Boulton](#) (Technical Editor)
Key Phrases: [meaningful markup](#), [multiple background images](#), [grey box method](#), [Internet Explorer](#), [Working Group](#), [Zen Garden](#) (more...)
★★★★★ (56 customer reviews)

List Price: \$54.99
Price: **\$34.64** & eligible for free shipping with **Amazon Prime**
You Save: **\$20.35 (37%)**

In Stock.
Ships from and sold by **Amazon.com**. Gift-wrap available.

Quantity: 1

[Add to Shopping Cart](#)

or

[Sign in](#) to turn on 1-Click ordering.

[Add to Wish List](#)

More Buying Choices

63 used & new from **\$22.00**

Have one to sell? [Sell yours here](#)



Multi-column Layouts

- Four column layouts can also be created, but often they will begin to look cluttered and are somewhat more difficult to design successfully.
- One of the best four-column designs that I know of can be found at <http://alistapart.com> which is a very well done design that looks very nice and clean.



A List Apart - Windows Internet Explorer

http://www.alistapart.com/

File Edit View Favorites Tools Help

Google Search

Share Check AutoFill Sign In

Favorites Suggested Sites Free Hotmail Web Slice Gallery KeepVid- Download and s...

A List Apart

ARTICLES • TOPICS • ABOUT • CONTACT • CONTRIBUTE • FEED

No. 323

FEBRUARY 8, 2011

Learn to love and make cool stuff with SVG and design web registration forms for kids.

Cross Platform Scalable Vector Graphics with svgweb

by **JIM RAY**

Pity Scalable Vector Graphics. It's been an official standard since before IE6 was released yet has never found much of an audience on the web, certainly not the one it deserves. Just as SVG was starting to establish some browser support, along came the canvas tag, stealing the thunder of dynamically generated client-side images. Yet despite all the attention being paid to canvas,

AN EVENT APART
coming to a city near you

EDITOR'S CHOICE
originally ran: July 20, 2010

SVG with a little help from Raphaël
by **BRIAN SUDA**

Want to make fancy, interactive, scalable vector graphics (SVGs) that look beautiful at any resolution and degrade with grace? Brian Suda urges you to consider Raphaël for your SVG heavy lifting.

Search ALA

GO

include discussions

Topics

- Code
- Content
- Culture
- Design
- Mobile
- Process
- User Science

Win a VIP Trip to SXSW
MAKE A GREAT FONT PAIR AND YOU'RE THERE!

http://www.alistapart.com/ Internet | Protected Mode: Off 100%



Multi-column Layouts

- One thing that the last three examples have illustrated, and you may also have noticed at web sites you normally visit, is that virtually every layout, regardless of the number of columns, has at the top a horizontal area, commonly known as a header, that spans the width of the page.
- The header's primary purpose is to present the brand identity of the site so users know which site they are on, but it is also frequently used for navigation elements, as the Amazon and A List Apart screenshots illustrate.



Multi-column Layouts

- Most pages also have a corresponding footer element that can provide navigation links once the user gets to the bottom of the page, and often hold copyright and legal disclaimers that need to appear on every page.
- The rest of the notes in this section are devoted to looking at different ways to produce multi-column layouts. Hopefully, these will give you some ideas for your own websites (your project) and you can use the basic framework as the basis for your webpages.



Width Matters

- As we progress through the different layouts, keep in mind that all of these layouts will increase in their vertical height automatically, according to the content within them.
- If you add more content, the layout increases its height to accommodate it, and that is **exactly** what you want to have happen.
- Controlling the horizontal width of the layout is the key to the way they function. Users detest horizontal scrolling, so it is important to be sure that they do not have to resort to it on your webpages.



Width Matters

- Most of the layout designs we will consider are based on using elements that are floated using CSS to create columns.
- These kinds of layouts can display incorrectly if they do not maintain key width dimensions. We'll explain all of this as we go along, but remember: You want to create layouts that expand vertically to accommodate any amount of content, without changing their width.
- Setting and controlling the width is the primary thing we'll be concerned with as we move along.



Floated Versus Absolutely Positioned Layouts

- There are two basic approaches to creating columns in your page layouts.
- You can float them side-by-side using the technique we illustrated in the previous section of the notes (see CSS-P – Part 3 and 4), or you can use absolute positioning and fix the width and location of the columns across the page.
- There are advantages and disadvantages to both techniques which we'll take a couple of pages to highlight:



Floated Versus Absolutely Positioned Layouts

- Floated columns are quick and easy to implement, but require that you be very careful to ensure that you don't accidentally cause the total width of the columns to exceed the width of the layout.
- For example, by increasing the width of a column by adding a large image to it. This would cause the right column (if the image were added to the left column) to be forced down and under the left column – clearly not a desirable effect. However, using the nested `<div>` technique we illustrated in the previous section of notes along with an `overflow` property, we can prevent this.



Floated Versus Absolutely Positioned Layouts

- Absolutely positioned columns do have a couple of notable advantages.
- First, you can sequence the markup to put the content first, or certainly earlier than it would occur using floated columns. Some people will claim that this can improve search engine visibility (personally, I think that referring links, good title tags, and judicious use of keywords in heading and content are the keys to being found by a search engine).



Floated Versus Absolutely Positioned Layouts

- Second, the columns stay in their specified locations under all circumstances, and the layout cannot “break” like a floated layout can.
- However, because the columns are absolutely positioned, they are removed from the normal flow and therefore have no sense of their relationship to one another.
- This means that it is extremely hard to create fluid layouts with absolute positioning. It also means that a footer will not get pushed down as content is added to the page, because the columns are independent and don’t interact with one another as they do in a floated layout.

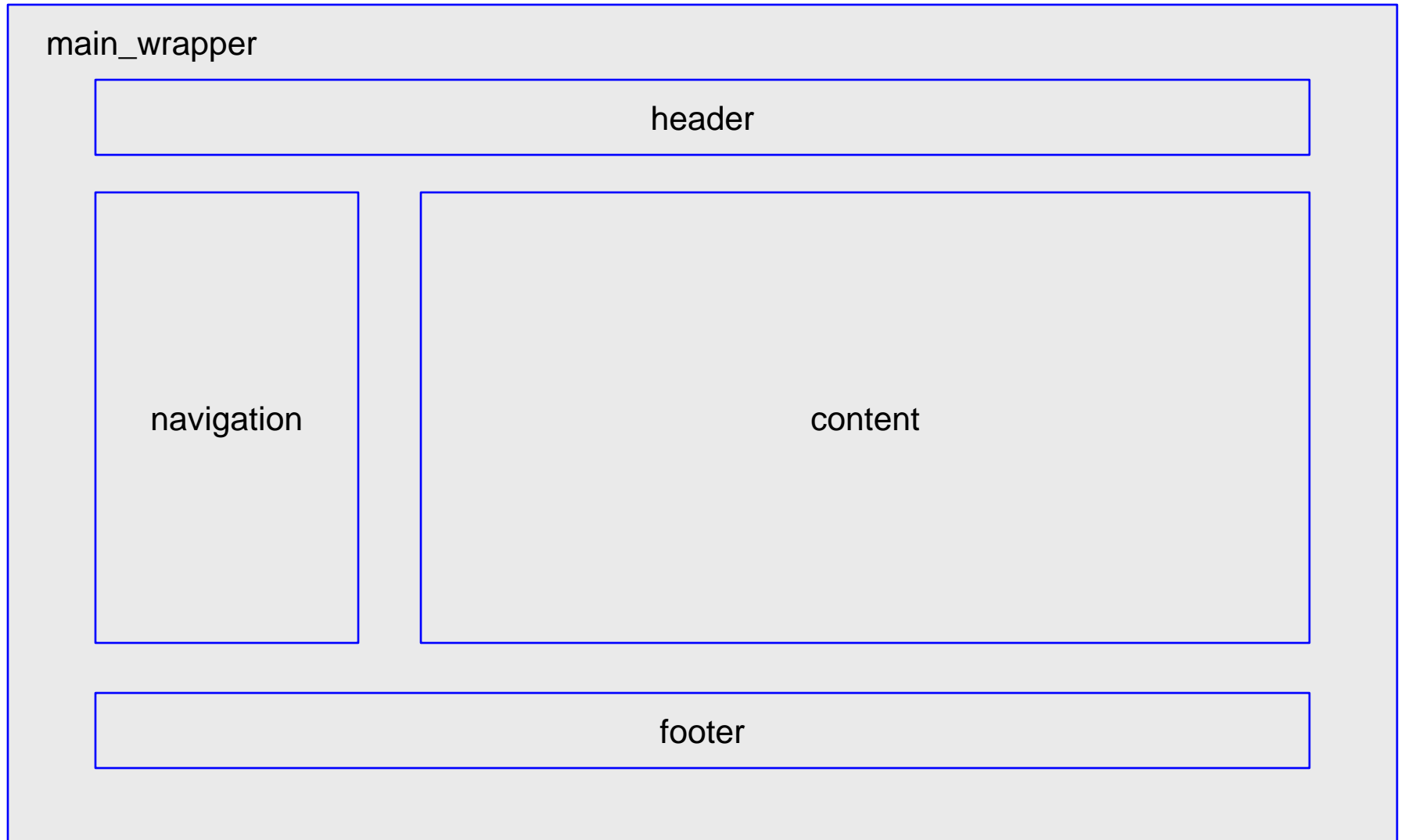


A Simple Two-Column Fixed-Width Layout

- As our first example of a basic page layout, we'll consider a simple two-column layout with a fixed width.
- This is a fairly common layout. It contains a relatively narrow left column for navigation elements and a wider right column that holds the page's content.
- In the example we'll build, both the left and right columns using a fixed width, and the layout will center itself in the browser window if the browser window is made wider than the width of the layout.
- The next page shows a storyboard for the two-column layout and the following page illustrates the rendered page.



A Simple Two-Column Fixed-Width Layout



A Simple Two-Column Fixed Width Layout

- [Nav item 1](#)
- [Nav item 2](#)
- [Nav item 3](#)
- [Nav item 4](#)
- [Nav item 5](#)

About This Layout

This page is styled with CSS. It demonstrates a float-based two-column layout with fixed width.

The Concept

The four structural <div> elements — header, nav, content and footer — nest inside a fixed width containing <div>. The two columns, nav and content, are sized with percentages and are floated so they sit side by side. The footer is cleared so it sits beneath whichever of the floated columns is longest.

Auto left and right margin settings are applied to the fixed-width containing <div>, which makes the layout center in a wide browser window.

The Files

This example uses two CSS files to style the page:

1. `two_column_fixed_widthCSS.css`
2. `text_n_colorsCSS.css`

The XHTML markup file is called:

- `simple two column layout.html`

Note: Inner <div> elements inside each of the four main <div> elements allow padding and borders to be applied without affecting the width of the main structural <div> elements.

© 2011 - a CSS-based simple two-column layout from *CIS 4004 - Web Based Information Technology - Spring 2011* by Dr. Mark Llewellyn



A Simple Two-Column Fixed-Width Layout

- Rather than put screen shots of the entire markup and CSS files for this layout, I've put both the XHTML and the two CSS files on the course website for you to look at and use. In these notes, I'll include various pieces of the markup and CSS to discuss how the layout is created.
- This is also the first example in the course that has used two separate stylesheets, so you might notice as the first thing that there are two `<link>` elements in the header. This is quite common, particularly as layouts get more complicated; to separate some of the styles. In this case I have separated the CSS into a stylesheet that deals only with text and colors, and a separate stylesheet that deals with the columns.



```
C:\Courses\CIS 4004 - Web Based IT\Spring 2011\code\Page Layouts\Basic Page Layout - Part 1\simple two column fixed width layout.htm...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
fan club 1.css assignment2part2.css simple two column fixed width layout.html
1 <?xml version="1.0" encoding="UTF-8" standalone="no"?>
2 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
4 <html xmlns="http://www.w3.org/1999/xhtml">
5 <head>
6 <title>Simple Two-Column Fixed Width Layout</title>
7 <link rel="stylesheet" href="text_n_colorsCSS.css" type="text/css" />
8 <link rel="stylesheet" href="two_column_fixed_widthCSS.css" type="text/css" />
9 </head>
10 <body class="beige">
11
12 <div id="main_wrapper">
13
14 <div id="header">
15 <div id="header_inner">
16 <h1>A Simple Two-Column Fixed Width Layout</h1>
17 </div><!-- end header_inner -->
18 </div><!-- end header -->
19
20 <!-- left column for navigation elements -->
```




```
C:\Courses\CIS 4004 - Web Based IT\Spring 2011\code\Page Layouts\Basic Page Layout - Part 1\simple two column fixed width layout.htm...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
fan club 1.css assignment2part2.css simple two column fixed width layout.html
19
20 <!-- left column for navigation elements -->
21 <div id="nav">
22   <div id="nav_inner">
23     <ul>
24       <li><a href="#">Nav item 1</a></li>
25       <li><a href="#">Nav item 2</a></li>
26       <li><a href="#">Nav item 3</a></li>
27       <li><a href="#">Nav item 4</a></li>
28       <li><a href="#">Nav item 5</a></li>
29     </ul>
30   </div><!-- end nav_inner -->
31   <!--  -->
32 </div><!-- end nav -->
33
34 <!-- right column for content -->
35 <div id="content">
36   <div id="content_inner">
37     <h2>About This Layout</h2>
38     <p><strong>This page is styled with CSS</strong></p>
39   </div>
40 </div>
Hyper Text Markup Lang nb char : 2826 nb line : 83 Ln : 31 Col : 83 Sel : 0 UNIX ANSI INS
```



```
C:\Courses\CIS 4004 - Web Based IT\Spring 2011\code\Page Layouts\Basic Page Layout - Part 1\simple two column fixed width layout.htm...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
fan club 1.css assignment2part2.css simple two column fixed width layout.html
65     </div><!-- end content_inner -->
66 </div><!-- end content -->
67
68 <!-- footer -->
69 <div id="footer">
70     <div id="footer_inner">
71         <p>&copy; 2011 - a CSS-based simple two-column layout from
72             <a href="http://www.cs.ucf.edu/courses/cis 4004/spr2011">
73                 <em>CIS 4004 - Web Based Information Technology - Spring 2011</em></a> by
74             </p>
75     </div><!-- end footer_inner -->
76 </div><!-- end footer -->
77
78 </div><!--end main wrapper-->
79
80 </body>
81 </html>
82
83
Hyper Text Markup Lang nb char : 2826 nb line : 83 Ln : 28 Col : 48 Sel : 0 UNIX ANSI INS
```



A Simple Two-Column Fixed-Width Layout

- In the body of the document there are four `<div>` elements each containing a nested `<div>` element. There is also a fifth `<div>` element that is used to style the entire page.
- The four `<div>` elements cover the header, navigation area, content, and footer.
- Recall that we utilized this nested `<div>` structure to allow us more flexibility for modifying the way columns look internally without modifying their enclosing container. We will begin to make extensive use of this technique as we examine these more complex layouts. (See CSS-P – Part 2, page 39.)



A Simple Two-Column Fixed-Width Layout

- Although these inner divs add a little extra markup, they will greatly simplify styling and modifying your layouts, as they solve all the commonly encountered box model problems of columns getting larger as you apply margins, borders, and padding.
- Using inner divs on the elements that have critical widths associated with them removes the problem of every time you add a 1 pixel border around that `<div>` needing to remember to subtract two-pixels off the stated width (left + right) in order to keep the `<div>`'s width constant.



A Simple Two-Column Fixed-Width Layout

- Now let's look at the CSS stylesheet that is used to style the columns. This is the file named: `two_column_fixed_widthCSS`.
- This CSS stylesheet contains only the definitions for the column areas of the layout. No font styles, colors, or any other presentation styling CSS rules occur in this stylesheet (there is one color styling of the right border for inner navigation column that was easier to do here than in the other stylesheet).



```
C:\Courses\CIS 4004 - Web Based IT\Spring 2011\code\Page Layouts\Basic Page Layout - Part 1\two_column_fixed_wid...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
fan club 1.css assignment2part2.css simple two column fixed width layout.html two_column_fixed_widthCSS.css
1 /* CSS Stylesheet for Simple Two Column Layout */
2 /* filename: two_column_fixed_widthCSS.css */
3
4 #main_wrapper {
5     width:840px; /* widths of columns will change proportionately as this width is changed */
6     margin-left:auto; /* centers layout in browser */
7     margin-right:auto; /* centers layout in browser */
8     text-align:left; /* resets the centering hack for IE6 on the body tag */
9 }
10 #header { text-align: center;
11 }
12
13 #nav {
14     width:22%; /* this width + content width must total 100% */
15     float:left; /* floats on nav and content divs make them sit side by side */
16 }
17 #content {
18     float:left; /* floats on nav and content divs make them sit side by side */
19     width:78%; /* this width + nav width must total 100% */
20     top:0px;
21 }
Casca nb char : 1426 nb line : 41 Ln:9 Col:2 Sel:0 Dos\Windows ANSI INS
```



```
C:\Courses\CIS 4004 - Web Based IT\Spring 2011\code\Page Layouts\Basic Page Layout - Part 1\two_column_fixed_wid...
File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ?
fan club 1.css assignment2part2.css simple two column fixed width layout.html two_column_fixed_widthCSS.css
21 }
22 #footer {
23     clear:both; /* makes the footer sit below whichever column is longest */
24 }
25 #header_inner, #nav_inner, #content_inner {
26     overflow:hidden; /* clips oversize elements that would otherwise expand divs and break the layout */
27 }
28 #header_inner {
29     padding:1em 2em; /* creates space between the box and the content */
30 }
31 #nav_inner {
32     padding:1em .8em; /* creates space between the box and the content */
33     border-right:3px solid black; /*was #B33 */
34 }
35 #content_inner {
36     padding:0 1em 1em 1.5em; /* creates space between the box and the content */
37 }
38 #footer_inner {
39     padding:.5em 1em; /* creates space between the box and the content */
40     text-align:center;
41 }
```

Casca nb char : 1426 nb line : 41 Ln: 9 Col: 2 Sel: 0 Dos\Windows ANSI INS



A Simple Two-Column Fixed-Width Layout

- Without this CSS stylesheet, the four main `<div>` elements would be stacked one above the other across the width of the browser by default (This is shown on the next page).
- The CSS that converts that default arrangement into the two column layout are the ids `header`, `nav`, `content`, and `footer`.
- The `nav` and `content` divs are floated, which will make them set side-by-side, and each has a percentage width applied to it that together totals 100%, so they are the same width as the header and the footer.



A Simple Two-Column Fixed Width Layout

- [Nav item 1](#)
- [Nav item 2](#)
- [Nav item 3](#)
- [Nav item 4](#)
- [Nav item 5](#)

About This Layout

This page is styled with CSS. It demonstrates a float-based two-column layout with fixed width.

The Concept

The four structural <div> elements — header, nav, content and footer — nest inside a fixed width containing <div>. The two columns, nav and content, are sized with percentages and are floated so they sit side by side. The footer is cleared so it sits beneath whichever of the floated columns is longest.

Auto left and right margin settings are applied to the fixed-width containing <div>, which makes the layout center in a wide browser window.

The Files

This example uses two CSS files to style the page:

1. [two_column_fixed_widthCSS.css](#)
2. [text_n_colorsCSS.css](#)

The XHTML markup file is called:

- [simple two column layout.html](#)

Note: Inner <div> elements inside each of the four main <div> elements allow padding and borders to be applied without affecting the width of the main structural <div> elements.

© 2011 - a CSS-based simple two-column layout from [CIS 4004 - Web Based Information Technology - Spring 2011](#) by Dr. Mark Llewellyn



A Simple Two-Column Fixed-Width Layout

- The footer has a `clear: both` rule to ensure that it sits below whichever of the two columns is the longest as determined by their content.
- Normal flow does the rest of the work for us – the header and footer are by default the full width of the `main_wrapper` containing element.
- As the name suggests, the `main_wrapper` div encloses the entire layout, and I've given it an arbitrary width of 840 pixels. This `<div>` element sets the overall width of the layout because it is the parent of the four `<div>` elements, and therefore its width determines the width of the footer, the header, and the combined width of the columns.



A Simple Two-Column Fixed-Width Layout

- There are two nice things about this arrangement.
- (1) By simply adjusting the width of the `main_wrapper` div, you can modify the overall layout width (the elements inside this div change their width proportionally, and their code does not need modification to do so).
- (2) Just by adding `auto` left and right margins to the `main_wrapper` div, the layout centers itself nicely in the browser window if the user make the browser window wider than the `main_wrapper` div; currently, 840 pixels.



A Simple Two-Column Fixed-Width Layout

- The following screen shot shows the page rendered using only the `two_column_fixed_widthCSS` stylesheet applied to the page (the stylesheet that sets the fonts, colors, etc. has not been applied in this case).
- Notice on page 31, that I created a single rule for all the inner `<div>` elements, using a group selector, which hides the overflow of oversized content from within them.

```
#header_inner, #nav_inner, #content_inner {  
    overflow:hidden;  
    /* clips oversize elements that would otherwise  
       expand divs and break the layout */  
}
```



A Simple Two-Column Fixed Width Layout

- [Nav item 1](#)
- [Nav item 2](#)
- [Nav item 3](#)
- [Nav item 4](#)
- [Nav item 5](#)

About This Layout

This page is styled with CSS. It demonstrates a float-based two-column layout with fixed width.

The Concept

The four structural `<div>` elements — header, nav, content and footer — nest inside a fixed width containing `<div>`. The two columns, nav and content, are sized with percentages and are floated so they sit side by side. The footer is cleared so it sits beneath whichever of the floated columns is longest.

Auto left and right margin settings are applied to the fixed-width containing `<div>`, which makes the layout center in a wide browser window.

The Files

This example uses two CSS files to style the page:

1. `two_column_fixed_widthCSS.css`
2. `text_n_colorsCSS.css`



A Simple Two-Column Fixed-Width Layout

- The CSS `overflow` property controls how element deal with content within them. The default setting , `visible`, causes the element to expand to enclose the content within it. For example, if you modify the markup and add a large image to the left column, that column would normally expand vertically and horizontally to display the entire image.
- Unexpected changes in column width like this are the curse of floated layouts, as the rightmost column will move down under the left column if the columns get pushed over to the right, and suddenly it doesn't have room to sit where it belongs using the relative positioning of the float.



A Simple Two-Column Fixed-Width Layout

- With the `overflow: hidden` rule applied, instead of getting larger, the column will retain its defined width and simply display the part of the image that does fit. The column will not change its size to accommodate the image.
- Of course, its good practice not to add an oversized element in the first place, but sometimes you might not have control over the content in the future.



A Simple Two-Column Fixed-Width Layout

- Using this nested `<div>` technique you can freely apply borders, margins, and padding to the inner divs. Because they do not have an explicit width, they always fill their respective column divs.
- I was able to add the black line down the right side of the navigation area by adding it to the inner nav div. I could do this without affecting the width of the critical “outer” div width and thereby breaking the layout.
- Just remember: Don’t apply visual styles directly to those main column `<div>` elements; style their related “inner” `<div>` elements instead.



A Simple Two-Column Fixed Width Layout

[Nav item 1](#)
[Nav item 2](#)
[Nav item 3](#)
[Nav item 4](#)
[Nav item 5](#)

About This Layout

This page is styled with CSS. It demonstrates a float-based two-column layout with fixed width.

The Concept

The four structural <div> elements — header, nav, content and footer — nest inside a fixed width containing <div>. The two columns, nav and content, are sized with percentages and are floated so they sit side by side. The footer is cleared so it sits beneath whichever of the floated columns is longest.

Auto left and right margin settings are applied to the fixed-width containing <div>, which makes the layout center in a wide browser window.

The Files

This example uses two CSS files to style the page:

1. [two_column_fixed_widthCSS.css](#)
2. [text_n_colorsCSS.css](#)

The XHTML markup file is called:

- [simple two column layout.html](#)

Note: Inner <div> elements inside each of the four main <div> elements allow padding and borders to be applied without affecting the width of the main structural <div> elements.

© 2011 - a CSS-based simple two-column layout from [CIS 4004 - Web Based Information Technology - Spring 2011](#) by Dr. Mark Llewellyn

