

Scalable 2-Column Layout using a left-floated <div>

Objective

We want a column on the left (sidebar) with a width specified in 'em' units* so that it expands/contracts as the browser's default font-size is increased/decreased.

We also want a page header at the top of the page and a footer at the bottom of the page. The footer should be positioned at the bottom of the viewport/screen when the page is shorter than the screen but it should remain tied to the bottom of the page when the page is longer than the screen.

Outline of Method

We use a left-floated div to hold the sidebar contents and a separate absolutely-positioned div to provide a background color for the sidebar. The background layer is positioned inside a wrapper layer and extends from the top with a height of 100%. We have to include some style rules to fix an IE6 problem that can prevent the background color extending all the way down to the footer in some circumstances.

* We are using 'em' units so the resulting layout can be used with the Menukit option that allows menus to be sized in 'em' units. You can change the units to 'px' if you prefer.

1. Create a local site for this tutorial.

Download the [support files](#) for this tutorial. Create an empty site and copy the support files into it. Start Dreamweaver and open the start page, *2colDivs_start.htm*.

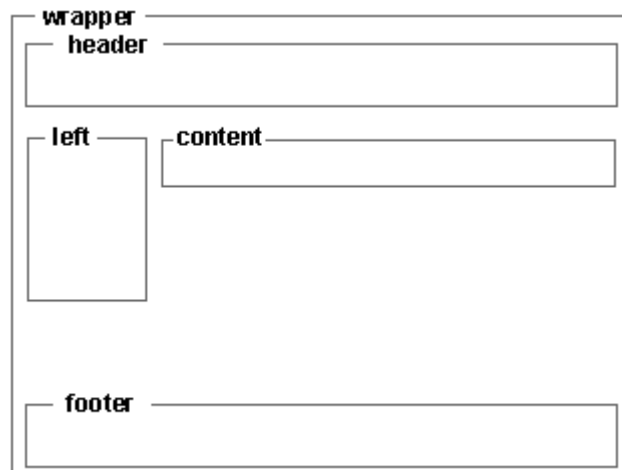
2. Analysis of the *2colDivs_start.htm* page.

The start page contains a labelled <div> for each of the four main areas: 'header', 'footer', 'left' and 'content' as shown in the diagram below.

The 'left' div is styled 'float: left' and is defined before the 'content' div in the HTML source file. This causes the 'content' div to appear on its right.

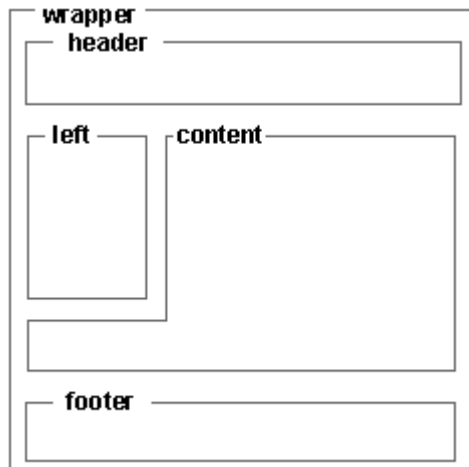
These 4 divs are enclosed in a 'wrapper' div. The 'wrapper' div allows us to keep the 'footer' div positioned at the bottom of the page when the page is longer than the screen.

The divs in *2colDivs_start.htm* have temporary borders and some filler text just to make them easy to see in a browser. We will delete these borders and text in a moment.



As text is added to the 'content' div, and/or the right side of the screen is dragged inwards, the text appears to flow downwards and around the floated 'left' div like this:

View *2colDivs_start.htm* in a browser if you want to see this effect. (Just ignore the behavior of the 'footer' for the moment.)

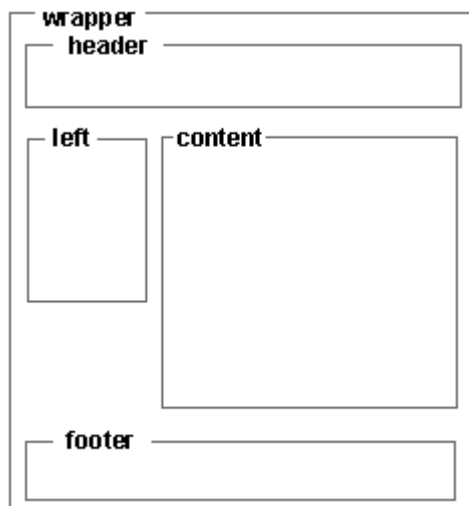


To prevent the text flowing under the 'left' div, **add a margin** to the left side of the 'content' div using a style like this:

You will find this line already present and commented out in *2colDivs_start.htm*. Just uncomment it.

```
#content {
margin: 0 0 0 7.5em;
...
}
```

The flow constraints imposed by this 7.5em margin on the left of 'content' give us the basic 2 column shape that we want:



The 'footer' div is absolutely positioned at the bottom of the 'wrapper' div.

The 'wrapper' div, which contains the 'footer' div, establishes a positioning context for the 'footer' div in this case because it is positioned relatively and has a height (actually min-height*) assigned to it.

The 'min-height: 100%' attribute ensures that 'wrapper' is *at least* as high as the screen but it can grow to the height of the page when the page is longer than the screen.

Annotated source code.

```

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<title>2 Column Scalable Layout (using a left-floated <div>)</title>
<style type="text/css">
html {
height: 100%;
}
body {
margin: 0;
padding: 0;
background: #FBFBFF;
color: #000033;
height: 100%;
}

#wrapper {
position: relative;
min-height: 100%;
}
#header {
height: 6em;
border: 1px solid blue;
}
#left {
float: left;
width: 7.5em;
border: 1px solid red;
}
#content {
/* margin: 0 0 0 7.5em; */
border: 1px solid orange;
}
#footer {
position: absolute;
bottom: 0;
height: 4em;
width: 100%;
border: 1px solid black;
}
</style>
</head>
<body>
<div id="wrapper">
<div id="header"> Header area in 2 column layout </div>
<div id="left"> Some text in the left column.
... </div>
<div id="content"> Some text in the content area.
... </div>
<div id="footer">Footer area in 2 column layout</div>
</div>
</body>
</html>

```

Browsers make the body element only as tall as is needed for the actual page contents - they do not stretch the height to fill the browser window (or more correctly, the viewport). But we can force the <body> to stretch to 100% of the viewport by giving it a height of 100% with a style specification. In some browsers, the <body> element gets its height from the <html> element rather than directly from the viewport. To cater for these browsers we also give the <html> element a height of 100%.

Note The '#wrapper' element includes the attribute 'position: relative'. That is required to make 'wrapper' the **positioning context** for 'footer'.

Why is this important?

(Does **not** apply to IE6)

If 'wrapper' were not the positioning context (eg. if its 'position' had been 'static' or omitted altogether) then the footer would be positioned at the bottom of the next outer containing block i.e. the <body> element in this case. Since <body> has been given a height of 100% (the height of the viewport), the footer would always appear at a fixed position on the page equal to the height of the viewport regardless of the actual size of the page - not what we want.

2colDivs_start.htm

3. Add a shim to keep the footer below the flowed content.

Since the footer is absolutely positioned it is removed from the document flow and does not affect where any of the flowed content is placed. In other words its presence is ignored. To see the effect of this, view the page in a browser and try dragging the bottom of the screen up and down.

Now *delete the temporary borders* (they are commented as 'temporary' in the source).

After you remove these borders you can make the browser's vertical scrollbar disappear completely by dragging the bottom of the screen downwards until it is below the flowed content. Now move the bottom of the screen upwards and notice that the vertical scrollbar becomes visible again as soon as the bottom of the screen (NOT THE FOOTER) touches the bottom of the flowed content in either column.

Make your browser window narrow enough so that the content in the main area is taller than the content in the left column.

This reveals a small problem that we must fix...

...the footer, being tied to the bottom of the screen, is also moved up and overlaps the lower part of the content. Actually, it overlaps the content to a height of 4em, the height of the 'footer' div.

A simple way to fix this is to insert a spacer div (a 'shim' in mechanical engineering terms) into the flow at the bottom of the 'wrapper' div. This spacer div must have the same height as the footer AND IT MUST BE FLOWED content i.e. its position attribute must have the value 'static'. Add the shim just before the footer div like this:

Note. 'static' is the default value of 'position' so we can just omit it from the stylesheet element.

```
vel lorem.</div>
<div class="shim"></div>
<div id="footer">Footer area in 2 column
layout</div>
</div>
```

Note. The 'footer' div is absolutely positioned and has no effect on flowed content inside the 'wrapper' div. Therefore it doesn't matter whether we place the shim (flowed) before or after 'footer' in the source code as long as we place it inside the 'wrapper' div.

Make the shim the same height as the footer:

```
.shim {
height: 4em;
}
```

Here we used a class selector rather an ID selector because we may need to use a similar shim elsewhere.

If you are not sure about how the shim works, just give it a temporary background color and you'll be able to see its effect as you drag the bottom of the screen up and down.

```
.shim {
height: 4em;
background: #CCCCCC; /* temporary background color */
}
```

At this stage, everything looks fine provided the main content area is longer than the left column. But if the left column becomes longer than the content area then we will again see the footer overlapping the bottom of the left column. We can fix this by inserting the attribute 'clear: left' in the shim's style element.

```
.shim {
height: 4em;
clear: left;
}
```

'clear: left' means that the browser must place the shim *below* any previously left floated element.

The effect of this is to make the browser move the shim down until it is underneath the floated div. (Don't forget to delete the shim's temporary background color if you have not done so already.)

Your page should now look like page *2colDivs_inter.htm* which you can find in the [support files](#).

4. Set background colors for header, content and footer areas.

We have already set the text and background colors for the main content area by setting them for the <body> element. Now we can apply colors to the header and footer like this:

```
#header {
height: 6em;
background-color: #000059;
color: #F2F2FF;
}
```

```
#footer {
position: absolute;
bottom: 0;
height: 4em;
width: 100%;
background: #80A7E0;
color: #000033;
}
```

5. Set background color for the left column.

The Challenge

The floated 'left' div that we use for the left column may or may not extend all the way down to the 'footer' div depending on the size of its contents. This means we cannot use its background color to create a full length column as a sidebar. Instead we will create another, absolutely positioned div whose height we can control.

The Solution

Insert an absolutely positioned div, which we will identify as 'leftbar', inside the 'wrapper' div and place it in the source code *before* the 'header' and 'left' divs like this :

```
<div id="wrapper">
<div id="leftbar"></div>
<div id="header"> Header area in 2 column layout </div>
<div id="left"> Some text in the left column. Vivamus nisl.
```

Add a style element for it:

```
#leftbar {
position: absolute;
top: 0;
left: 0;
height: 100%;
width: 7.5em;
background-color: #DDEEFF;
```

```
color: #000033;
border-right: 1px solid #000055;
}
```

If you preview the page now with a Mozilla-based browser (don't use IE6 just yet) you will see that the 'left' column and part of the 'header' have disappeared. In fact, they have been hidden by the new 'leftbar' div which is now covering the 'left' column and part of the 'header'.

What has happened is that a new positioning context, with a higher stacking order on the Z-axis (coming out of or perpendicular to the screen), was created for the new div.

We can, however, bring the contents of 'left' back into view by creating a positioning context for 'left' which is higher than 'leftbar' on the Z-axis.

Why is the 'footer' not hidden as well?

Since 'footer' is positioned absolutely, it too is a positioning context and since it is defined in the source file after 'leftbar', it is given a higher stacking order than 'leftbar'.

Create a new positioning context for the 'left' div by inserting an attribute 'position: relative' into its style element. And since the 'left' div is located in the source code *after* the 'leftbar' div it receives a higher stacking value and is therefore placed on top of the background provided by 'leftbar' - exactly what we want.

```
#left {
position: relative;
float: left;
width: 7.5em;
}
```

Similarly, create a new positioning context for the 'header' div to bring it too in front of the sidebar by inserting an attribute 'position: relative' into its style element.

```
#header {
position: relative;
height: 6em;
background-color: #000059;
color: #F2F2FF;
}
```

Although the page renders well in standards-compliant browsers, it can cause a problem in Internet Explorer 6 in certain circumstances.

6. IE6 Background coloring problem.

In certain circumstances when a vertical scrollbar is required to view the lower section of a long page, the background color can disappear from the left column. The problem manifests itself when the page is longer than the screen *by more than the height of the footer* (in other words, if there is more of the page than just the footer scrolled down). The solution is in 2 parts:

Part 1

The first part takes care of the situation when the left column is longer than the content area. All we have to do here is add the background color and right border to the style for the 'left' div like this:

```
#left {
position: relative;
float: left;
```

```
width: 7.5em;
background-color: #DDEEFF;
color: #000033;
border-right: 1px solid #000055;
}
```

The above amendment does not adversely affect any other browser so we do not need to isolate it for IE.

Part 2

The problem of the missing background-color can still occur, however, if the content area is longer than the left column. To cater for that situation we add a wide border (same width as the left column) to the left side of the content area and give it the same color as the column background. Having added the wide border, we no longer want the wide margin on the left side of 'content' that we added earlier, so we redefine the left margin to have zero width. In this case, we will use a conditional comment to isolate this solution for IE6.

```
<!--[if lte IE 6]>
<style type="text/css">
#content {
border-left: 7.5em solid #DDEEFF;
margin-left: 0 !important;
}
</style>
<![endif]-->
```

Now when we view a long page in IE6 we see that the background-color does indeed extend down to the footer.

But the 1-pixel border is still missing on the bottom section of the page that scrolls up into view. To fix that we can include a 1-pixel background image in the 'content' div at position 7.5em along the x-axis and repeat it down the y-axis. Having positioned this single-pixel image at 7.5em we can expect the border to remain fixed in position at the boundary between the left column and the main content area.

Alas, there is yet another problem with IE6...

Whenever we change the default font-size, it appears that IE 6 forgets to recalculate the size of our 7.5em wide border. Fortunately, by including 'font-size: 100%;' (which should do nothing) in the #content style it seems we can remind IE 6 to do the required recalculation whenever the font-size changes.

So the final version of the style element that is needed to help IE6 looks like this:

```
<!--[if lte IE 6]>
<style type="text/css">
#content {
font-size: 100%;
border-left: 7.5em solid #DDEEFF;
margin-left: 0 !important;
background: url(borderpx.gif) 7.5em 0 repeat-y;
}
</style>
<![endif]-->
```

Your finished 2-column page should now look like page [2colDivs_final.htm](#) which you can find in the [support files](#).

Now you can delete the temporary filler text from the various divs to make an empty 2-column layout, *2colDivs.htm* ready to receive content.

Supplement

In order to illustrate the points we made earlier about IE6 we will now reduce the width of the layout to 95% and center it within dark (#999999) margins.

7. Reduce width of wrapper

After we reduce the width of the wrapper and center it, the page body background will become visible at the sides of the wrapper. Currently the content area inherits its text and background colors from the body element.

We need to transfer the current text and background colors of the body into the wrapper and change the color of the page background to the dark color that we want for the margins, like this:

```
body {
margin: 0;
padding: 0;
height: 100%;
background: #999999;
height: 100%;
}
#wrapper {
position: relative;
min-height: 100%;
background: #FBFBFF;
color: #000033;
}
```

Now we set the width of the wrapper to 95% and set its left and right margins to 'auto' to center it within the body:

```
#wrapper {
position: relative;
min-height: 100%;
background: #FBFBFF;
color: #000033;
width: 95%;
margin: 0 auto 0 auto;
}
```

If you view the page now in IE6 you will see that the footer is no longer tied to the bottom of the window.

It no longer looks right in IE6

8. What has happened to IE6 and how can we fix it?

Since it now has a width specification, the 'wrapper' div has become a positioning context and the footer is positioned absolutely at bottom.

IE6 considers the 'wrapper' div to have no height specification (it does not understand min-height) and so it determines the height of 'wrapper' from its contents.

If we set the height of 'wrapper' to 100% then it will inherit the height of its parent, the browser

window. In fact, it happens that IE6 interprets 'height' as 'min-height' and as a result, the height of 'wrapper' will increase to the length of the page when the page is longer than the screen.

But there is a potential problem.

If we set the height of 'wrapper' to 100% then it will break the solution we already arrived at above for browsers that implement the standard interpretation of 'height'.

So, we enclose the 'height' style in a conditional comment so it can be seen only by IE6:

```
<!--[if IE 6]>
<style type="text/css">
#wrapper {
height: 100%;
}
</style>
<![endif]-->
```

Your page should now look like page *2colDivs_centered.htm* which you can find in the [support files](#).