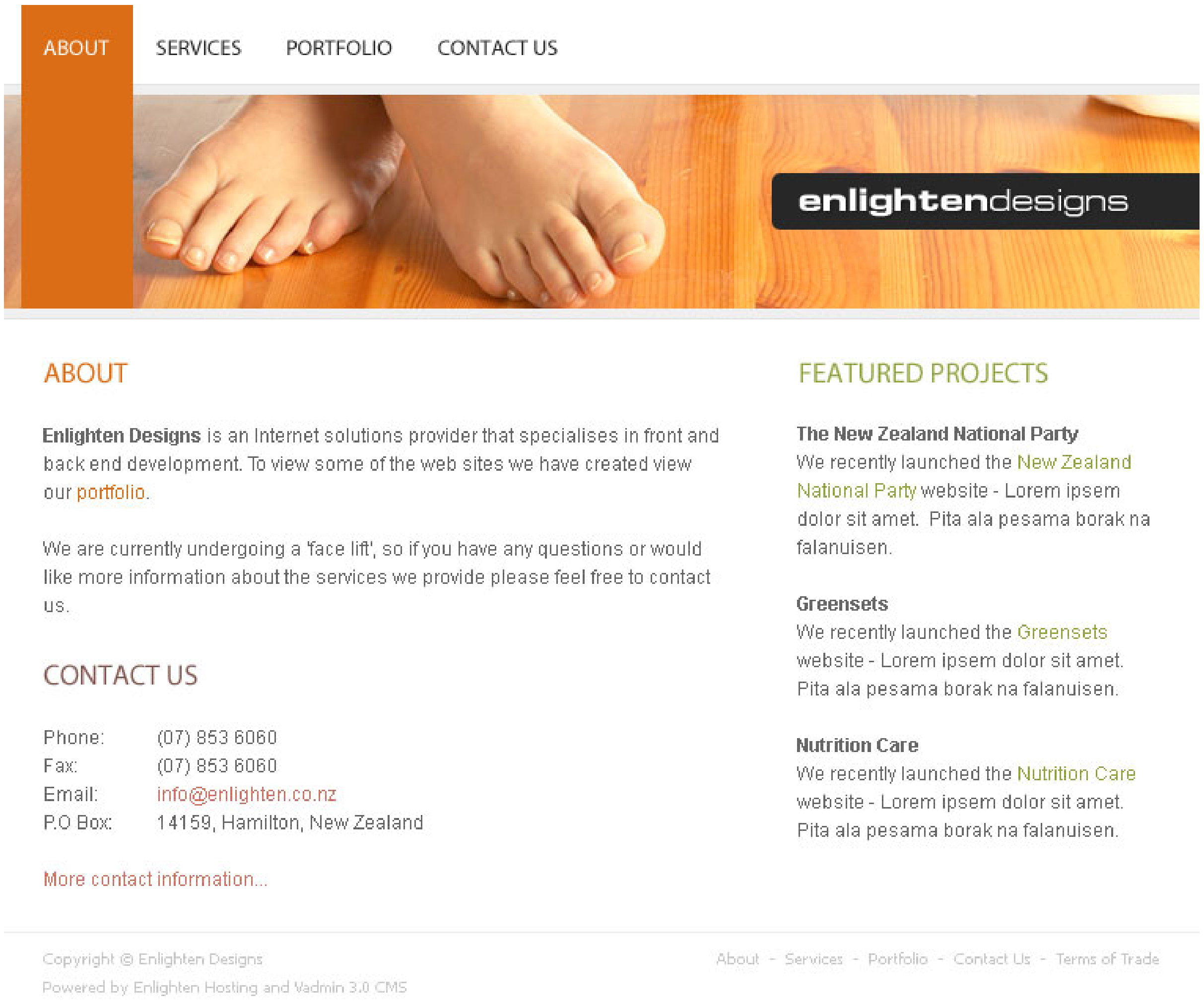
Using CSS to control layout

This guide will attempt to take you step by step, through the process of creating a fully functioning CSS layout.

Below is an image of what we will achieve once the tutorial is complete.



ABOUT

Enlighten Designs is an Internet solutions provider that specialises in front and back end development. To view some of the web sites we have created view our portfolio.

We are currently undergoing a 'face lift', so if you have any questions or would like more information about the services we provide please feel free to contact us.

CONTACT US

Phone:	(07) 853 6060
Fax:	(07) 853 6060
Email:	info@enlighten.co.nz
P.O Box:	14159, Hamilton, New Zealand

More contact information...

FEATURED PROJECTS

The New Zealand National Party
We recently launched the New Zealand National Party website - Lorem ipsem dolor sit amet. Pita ala pesama borak na falanuisen.

Greensets
We recently launched the Greensets website - Lorem ipsem dolor sit amet. Pita ala pesama borak na falanuisen.

Nutrition Care
We recently launched the Nutrition Care website - Lorem ipsem dolor sit amet. Pita ala pesama borak na falanuisen.

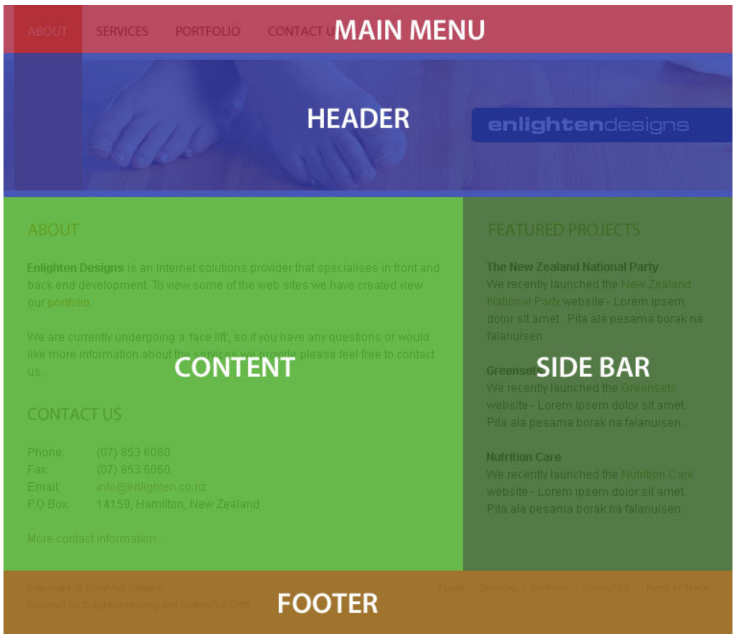
Copyright © Enlighten Designs
Powered by Enlighten Hosting and Vadmin 3.0 CMS

About - Services - Portfolio - Contact Us - Terms of Trade

First we need to identify the main structural elements of the design, so that we know how to structure our HTML document.

The web is very heavily based around rectangles, and we need to remember that when dividing up our design. These main divisions we make will end up being <div> tags. A <div> is basically a rectangular container that we can position using CSS.

The diagram below shows how we will divide the design.



We have identified 5 major elements:

- **Main Navigation**
  The primary navigation for this website. The images will change on hover (when the mouse cursor is on top of it).
  Width: 760px
  Height: 50px

- **Header**
  The website header includes a background image (purely for aesthetics), and the company name.
  Width: 760px
  Height: 150px

- **Content**
  The bulk of the website's content will go here.
  Width: 480px
  Height: Changes depending on content

- **Sidebar**
  This will have second-tier content that isn't as important as the main content.
  Width: 280px
  Height: Changes depending on content

- **Footer**
  Copyright information, credits, and an alternative text navigation.
  Width: 760px
  Height: 66px

This site will also be centered in the browser window. We now have all the info we need to start.

Starting off

Copy and paste the code below into notepad.

You will be creating all your styles on an external stylesheet for this tutorial.

As you can see from the code you will be using "@import" to import your style sheet.

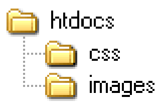
Meta Tag: A special HTML tag that provides information about a Web page. Unlike normal HTML tags, meta tags do not affect how the page is displayed. Instead, they provide information such as who created the page, how often it is updated, what the page is about, and which keywords represent the page's content. Many search engines use this information when building their indices.

http://www.webopedia.com/TERM/M/meta_tag.html 17/12/12

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
        "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html>
<head>
        <meta http-equiv="Content-type" content="text/html; charset=UTF-8"
/>
        <title>CompanyName - PageName</title>
        <meta http-equiv="Content-Language" content="en-us" />

        <meta name="description" content="Description" />
        <meta name="keywords" content="Keywords" />

        <meta name="author" content="Enlighten Designs" />

        <style type="text/css" media="all">@import
"css/master.css";</style>  <style type="text/css" media="all">@import
"css/master.css";</style>
</head>

<body>

</body>
</html>
```

Save this as index.html in your websites root (htdocs) directory.

The structure of your website directories should be like so:



Setting up the page

As you'll notice in the design, everything on our page is 760px wide or less, and nothing floats outside that width. What we are going to do is create a container for our page that is 760px wide, and centred in the middle of the page. Our 5 main elements will be placed inside this container.

Between the <body> </body> tags, create a <div> with an id="page-container" attribute:

```
<body>
<div id="page-container">
Hello world.
</div>
</body>
```

And that's all the HTML we need for our container. Onto the CSS.

Create a new blank text file, and save it as master.css in the /css/ directory.

Create a new rule in the stylesheet to select the page-container:

```
#page-container {


}
```

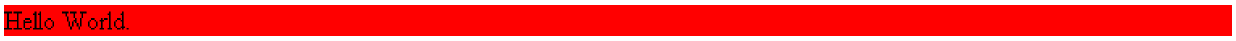
The # in front of the id tells the browser that we are selecting an id. For a class we would use a . instead e.g.: .page-container {}.

An id is a unique identifier that we use for things that are only going to occur once on the page. So for headers, footers, navigation, etc we use id's, and for any reoccurring elements like links we should use classes, which can occur multiple times on the same page.

We won't be able to see the changes we are making to this <div>, because it is transparent by default. So the first thing we will do is make the background of the div red, to give us a visible indicator of what we are doing:

```
#page-container {
background: red;
}
```

You should see something like this across the full width of your browser:



Hello World.

First we should set a width of 760px on this div.

Refresh the page in your browser to see the rule being applied.

Next we want to centre this div. This is done by setting the margins on it to auto. When the left and right margins are set to auto, they will even each other out and the div will sit in the centre of its container.

```
#page-container {
width: 760px;
margin: auto;
background: red;
}
```

Now you should have a centered red 760px wide block with “Hello World.” written in it. But its sitting about 8px away from the top/sides of the browser.

This is because the html and body tags have default margins and/or padding on nearly all browsers. So we need to write a CSS rule to reset the margins and padding on the html and body tags to zero. Add this rule to the very top of your css file:

```
html, body {
margin: 0;
padding: 0;
}
```

A comma in between CSS selectors stands for “or”, so here the rule will be applied to the html tag or the body tag. Because both exist on the page, it will be applied to both.

Brilliant, now our box is where it should be. Note that as more content is added to this div, it will automatically change its height to fit whatever content is placed inside it.

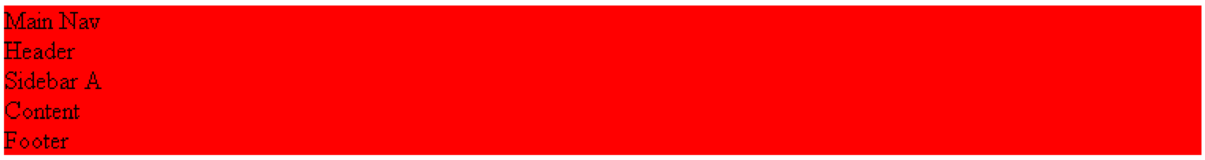
Let’s move on.

Creating the DIVs

We need to add 5 divs, all with individual id’s that describe their purpose. These divs will correspond to the major areas of the design we identified in Step 2. Replace the Hello World. text with the div’s below. Just for now we’ll also put text inside the divs for easy visual identification when we view the page.

```
<div id="page-container">
<div id="main-nav">Main Nav</div>
<div id="header">Header</div>
<div id="sidebar-a">Sidebar A</div>
<div id="content">Content</div>
<div id="footer">Footer</div>
</div>
```

Your page should now look something like this:


Main Nav
Header
Sidebar A
Content
Footer

Without CSS, the divs will be arranged from top to bottom, one under the other, in the same order as they are in the code. This is usually referred to as the ‘flow’ of the document.

So let’s use the information we have to make our divs the correct size.

Remove the red background from the #page-container, and add a new rule for #main-nav. Set the background of the main nav to red so we can see it, and set its height to 50px:

```
#page-container {
width: 760px;
margin: auto;
}
#main-nav {
background: red;
height: 50px;
}
```

Notice we didn't specify the width of the div. This is because by default, a div will stretch to fill its parent container, which in this case, is our #page-container div that was set to 760px wide.

Do the same thing for the header div, using the height we got in step one. Make this one blue.

```
#header {
background: blue;
height: 150px;
}
```

While we're at it, let's do the footer. The footer can be orange.

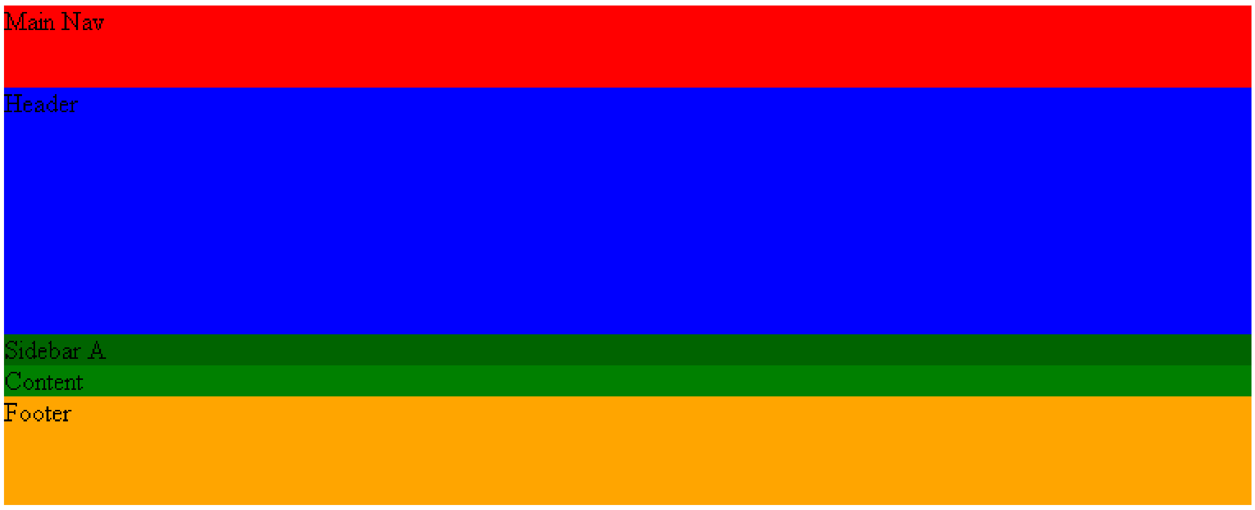
Remember when writing your stylesheet, that you should group your styles together. So all the header styles would go together. All the footer styles would be together, and all the navigation styles would be together. Also I find it helps to structure them in a similar order to the HTML, so header near the top, footer near the bottom.

```
#footer {
background: orange;
height: 66px;
}
```

Now the next 2 divs are slightly different. The heights are dependent on the content that's inside them, so we won't set a height. Let's make them dark green, and green. Put the rules in between the header and the footer rules in the css. This makes them easier to find once the stylesheet gets larger.

```
#header {
background: blue;
height: 150px;
}
#sidebar-a {
background: darkgreen;
}
#content {
background: green;
}

#footer {
background: orange;
height: 66px;
}
```

If all has gone well, you should have a page that looks like this:

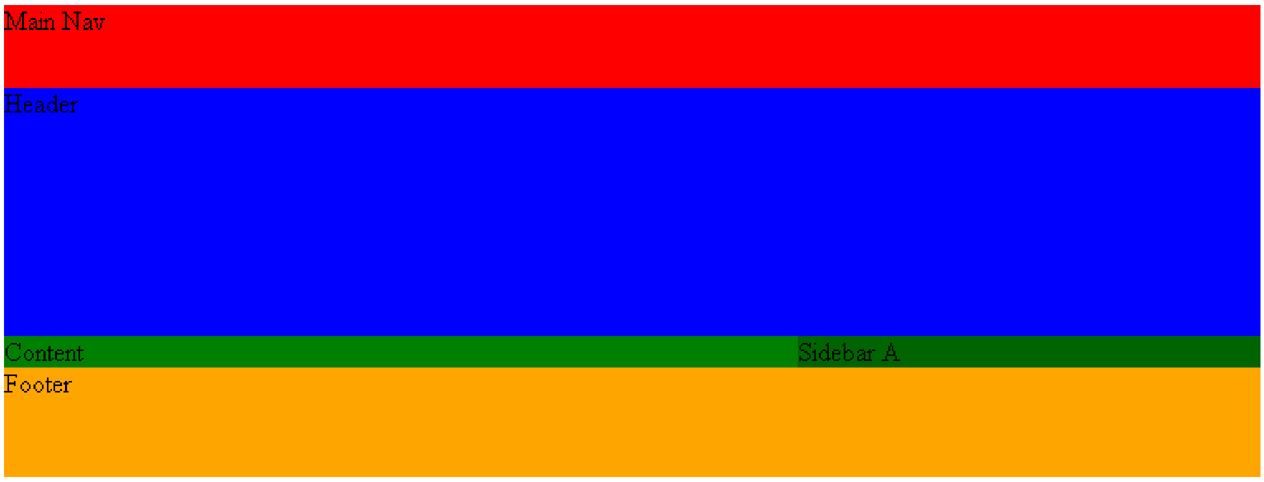


Floats can be a tricky concept to get your head around. Basically a float is an element that is aligned against the left or right side of its container.

In the case of this website, we are going to float our sidebar-a div to the right, with a width of 280px. Add the following to your CSS:

```
#sidebar-a {
float: right;
width: 280px;
background: darkgreen;
}
```

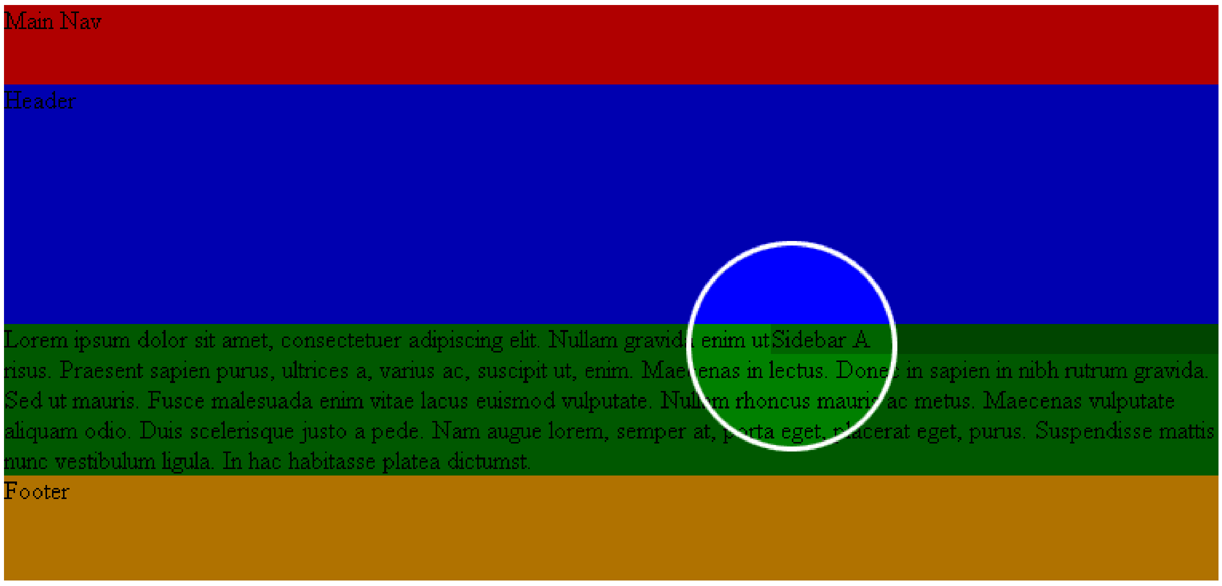
You have now successfully floated your first div, and you should now have a page that looks like this:



Just for testing purposes, replace the text in the content div to this:

```
<div id="content">

Add some text here of at least 10 lines.


</div>
```

Notice that the text in the content div wraps around the floated sidebar div, as shown below:



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam gravida enim ut risus. Praesent sapien purus, ultrices a, varius ac, suscipit ut, enim. Maecenas in lectus. Donec in sapien in nibh rutrum gravida. Sed ut mauris. Fusce malesuada enim vitae lacus euismod vulputate. Nullam rhoncus mauris ac metus. Maecenas vulputate aliquam odio. Duis scelerisque justo a pede. Nam augue lorem, semper at, porta eget, placerat eget, purus. Suspendisse mattis nunc vestibulum ligula. In hac habitasse platea dictumst.

This isn't what we want. We want the content div to sit along side the sidebar div, with its right edge against the left edge of the sidebar.

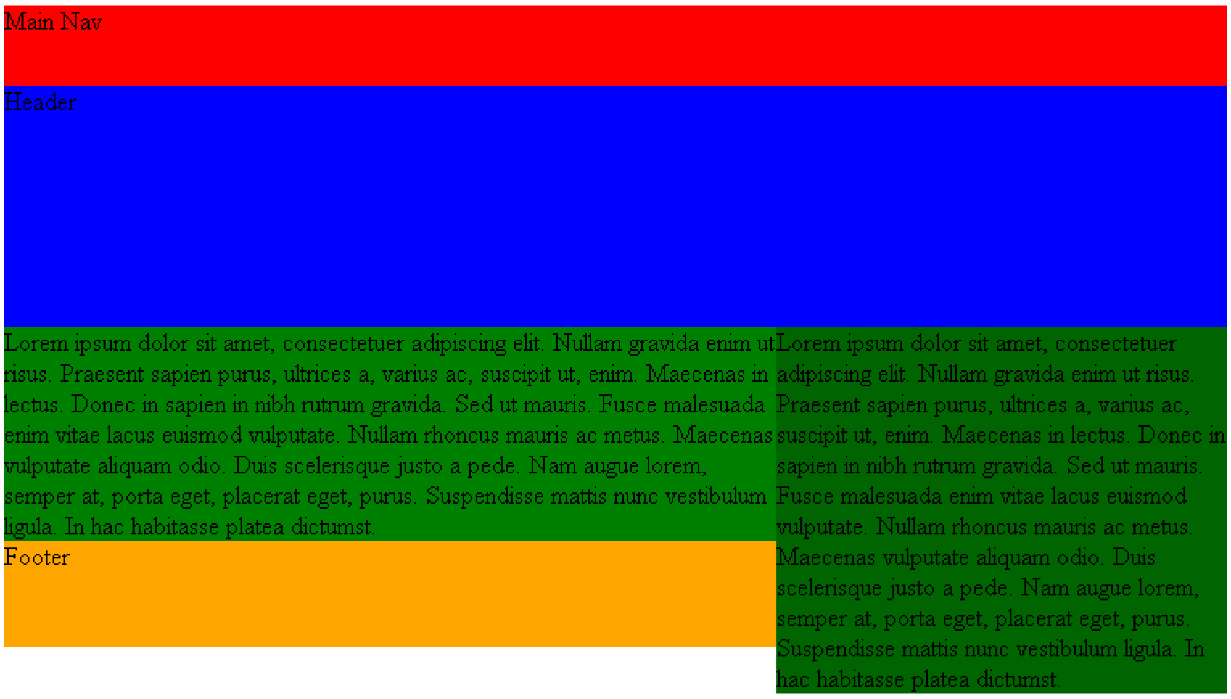
An easy way to achieve this in a float layout like this, is to put a right margin on our content div that is the same width as our sidebar, in this case 280px. This will push the right edge of the content away from the right edge of the page-container.

```
#content {
margin-right: 280px;
background: green;
}
```

Great, we've almost got the float layout sussed. But there's one more thing we need to consider… what happens if the sidebar div is taller than the content div?

Let's see. Copy and paste this text into the sidebar div:

```
<div id="sidebar-a">

Add text here so that so that the container becomes longer then the
content container as shown in the screenshot below.



</div>
```



Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam gravida enim ut risus. Praesent sapien purus, ultrices a, varius ac, suscipit ut, enim. Maecenas in lectus. Donec in sapien in nibh rutrum gravida. Sed ut mauris. Fusce malesuada enim vitae lacus euismod vulputate. Nullam rhoncus mauris ac metus. Maecenas vulputate aliquam odio. Duis scelerisque justo a pede. Nam augue lorem, semper at, porta eget, placerat eget, purus. Suspendisse mattis nunc vestibulum ligula. In hac habitasse platea dictumst.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam gravida enim ut risus. Praesent sapien purus, ultrices a, varius ac, suscipit ut, enim. Maecenas in lectus. Donec in sapien in nibh rutrum gravida. Sed ut mauris. Fusce malesuada enim vitae lacus euismod vulputate. Nullam rhoncus mauris ac metus. Maecenas vulputate aliquam odio. Duis scelerisque justo a pede. Nam augue lorem, semper at, porta eget, placerat eget, purus. Suspendisse mattis nunc vestibulum ligula. In hac habitasse platea dictumst.

That's not what we want at all. The reason the footer hasn't moved down is because the sidebar is 'floated' right.

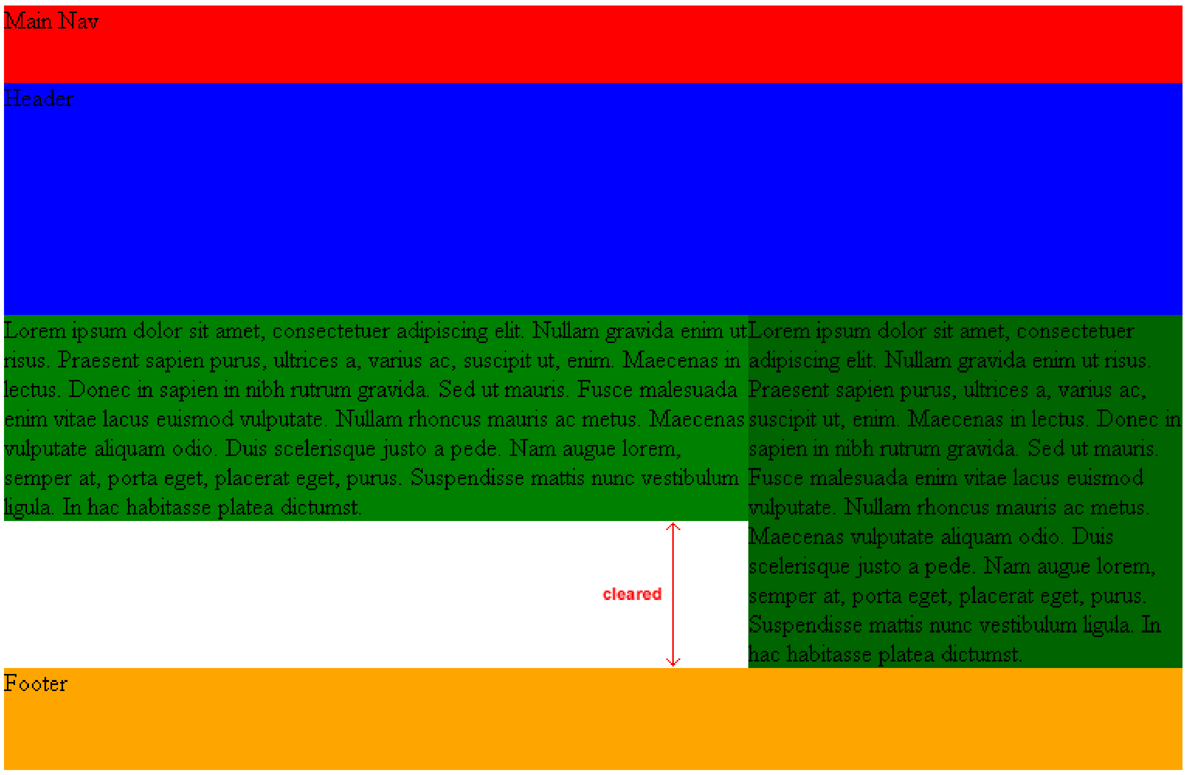
Explanation: By default, any floated element will not push down elements that are below it. This is because floated elements are not considered part of the document 'flow'. It's like they are on another layer 'floating' above the other elements, and because of this, it can't effect their positions.

What can we do to fix this problem? Introducing the “clear” css property.

Add this to your stylesheet:

```
#footer {

clear: both;

background: orange;

height: 66px;

}
```

When an element has the clear property assigned, if it comes into contact with a float it is placed right below where that float ends. You can specify if it is effected by only left floats or only right floats, in this case we could use either ‘right’ or ‘both’. We’ll use clear: both just to be safe.



Main Nav

Header

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam gravida enim ut risus. Praesent sapien purus, ultrices a, varius ac, suscipit ut, enim. Maecenas in lectus. Donec in sapien in nibh rutrum gravida. Sed ut mauris. Fusce malesuada enim vitae lacus euismod vulputate. Nullam rhoncus mauris ac metus. Maecenas vulputate aliquam odio. Duis scelerisque justo a pede. Nam augue lorem, semper at, porta eget, placerat eget, purus. Suspendisse mattis nunc vestibulum ligula. In hac habitasse platea dictumst.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Nullam gravida enim ut risus. Praesent sapien purus, ultrices a, varius ac, suscipit ut, enim. Maecenas in lectus. Donec in sapien in nibh rutrum gravida. Sed ut mauris. Fusce malesuada enim vitae lacus euismod vulputate. Nullam rhoncus mauris ac metus. Maecenas vulputate aliquam odio. Duis scelerisque justo a pede. Nam augue lorem, semper at, porta eget, placerat eget, purus. Suspendisse mattis nunc vestibulum ligula. In hac habitasse platea dictumst.

Footer