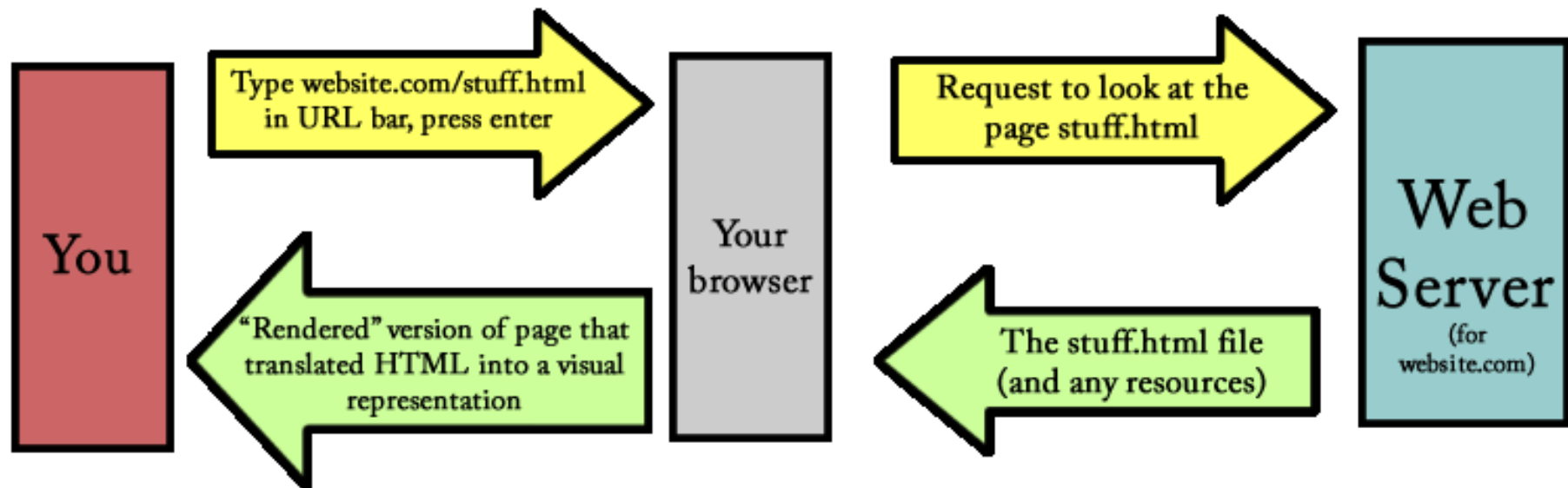


# Intro to Web Design

ACM Webmonkeys @ UIUC

# How do websites work?



Note that a similar procedure is used to load images, etc.

# What is HTML?

- An HTML file is just a plain text file.
  - You can write all your HTML in Notepad or gEdit
  - Similar to .cpp, .tex, .java, etc
- **HyperText Markup Language**
  - The word "markup" means it only indicates *how* the page should look and behave
  - Unlike a programming language, does not contain explicit instructions for the browser
    - "client-side" languages like Javascript, however, do
- Uses an XML-derived format

# The key idea behind XML (and HTML)

- Content is organized into **ELEMENTS** -- essentially, a container
- Elements specify a part of the webpage, and can alter how their content is displayed
- For example:

Sometimes you want **bold** text.

Sometimes you want `<b>bold</b>` text.

You should visit [this page](http://www.example.com).

You should visit `<a href='http://www.example.com'>this page</a>`.

# More about elements

- Note that each element has a **starting** tag and a **closing** tag.
  - Each tag is surrounded by angular braces (<>)
  - Starting tags have the name of the tag, plus any attributes to qualify the tag
    - list attributes one after the other, in the form x="y" z="a" b="c d"
  - Ending tags have a forward slash (/) before the tag name, and **cannot** have attributes.

```
<tagname attr="value" something="else">
```

```
Content goes here. You can <nest>tags</nest>.
```

```
</tagname>
```

# Anatomy of an element

the opening (or starting) tag



attributes



```
<element name attr="val" foo="bar">
```

contents

```
</element>
```



the closing tag



# Commonly-used elements

`<b>text</b>`      *Makes the contained text bold.*

`<i>text</i>`      *Italicizes the contained text.*

`<u>text</u>`      *Underlines the contained text.*

`<a href="http://www.google.com">Link Text</a>`

*Turns the contents into a link to the URL specified in the "href" attribute.*

``

*Inserts an image. The path can be a full URL, or a relative path (that is, relative to the HTML file).*

`<br/>`      *Inserts a line break.*

# General structure of a basic HTML page

`<html>` (the container element that should wrap everything)

`<head>` (this section is **ONLY** for information for the browser)

`<title>My Webpage</title>`

`</head>`

`<body>` (this is where your actual content goes)

`</body>`

(this be dead man's land)

`</html>`



# So, let's make a website.

- On the next slide is a mock-up (i.e., a picture of what we want it to look like) of a website a client wants.
- Rather than list tags one by one, let's walk through the basics by seeing what we need to know in order to make this website
- Feel free to follow along by yourself:
  - Open up Notepad, gEdit, or whatever
  - Create a new file and save it as example.html
    - In Notepad, make sure you don't save it as example.html.txt instead
  - Right click on the file (in Explorer or Nautilus) and open with Firefox (or Chrome) to view it.

# My Blog



This is a sidebar.  
Lorem ipsum dolor  
sit amet, consete  
tur adipisicing elit,  
sed do eiusmod  
tempor incididunt ut  
labore et dolore  
magna aliqua.

Ut enim ad minim  
veniam, quis  
nostrud exercitation  
ullamco laboris nisi  
ut aliquip ex ea com  
modo consequat.

## Dolorem ipsum


Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

# Where to start?

- I recommend working from the top down. Browsers will render your HTML from the top down as well, so it makes sense.
- However, to make things a bit easier, first consider how everything will fit together.
  - In this case, we want to structure the layout of the page
  - There are multiple ways to do this, but let's start with the simplest and oldest: a table.

# Tables

- In this context, "table" refers to the sorts of tables you see used to display statistics, poll results, etc.
- So why are we using it to structure the layout of our page?
  - Because looked at in the right way, the page actually fits into a table well (provided we can merge some cells):

My Blog	
 <p>This is a sidebar. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.</p> <p>Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.</p>	<h2>Dolorem ipsum</h2> <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum</p>

# How to make a table

- For our table:
  - First, we need to lay out the basic 2-col, 2-row structure.
  - Then, we merge the top two cells to make the title bar.
- In HTML:
  - First, use the `<table>` tag to wrap the entire table.
  - Use the `<tr>` (table row) tag for each row.
  - Within each row, use the `<td>` (table data) for each cell.
- So, we get:

# Our code so far:

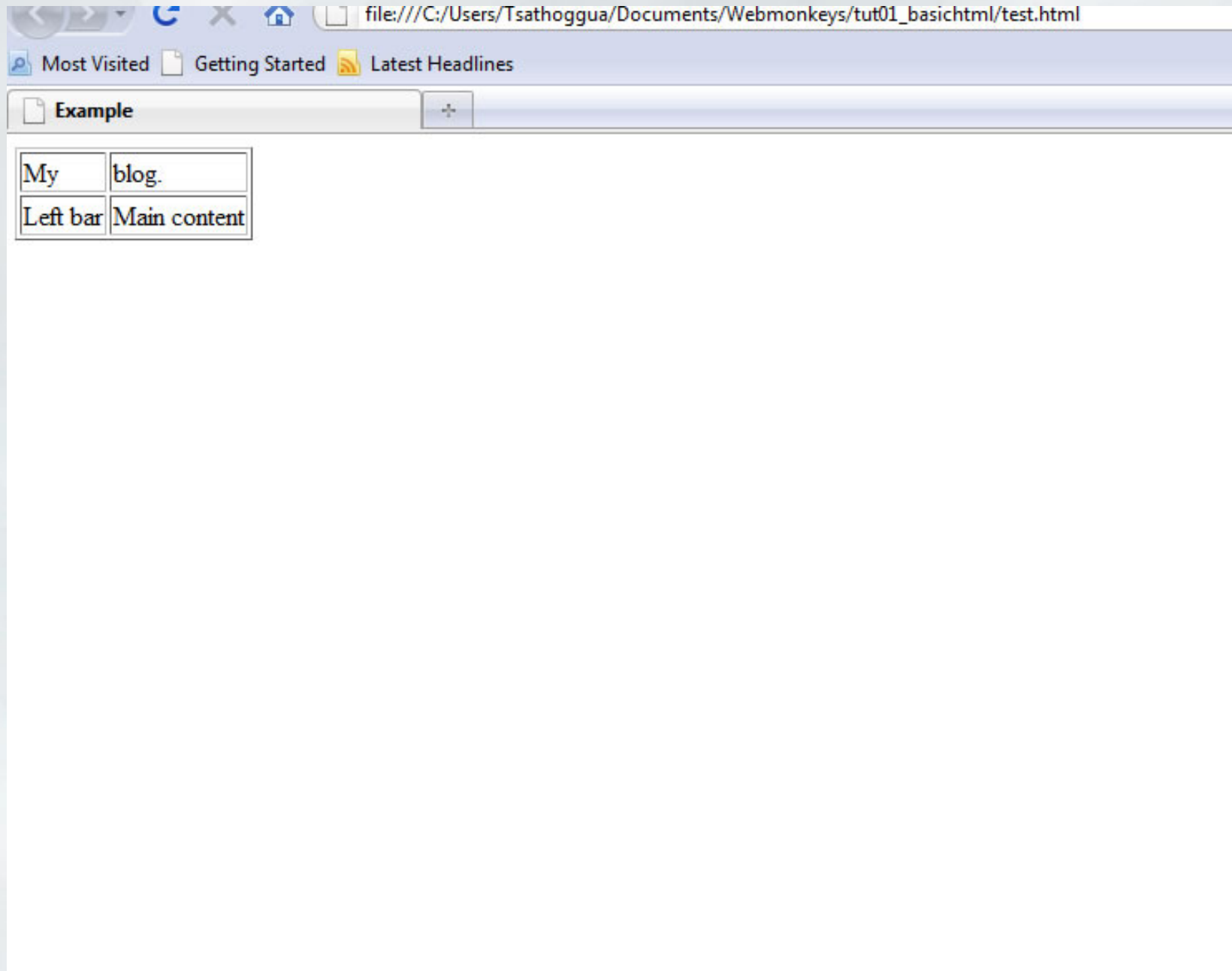
```
<html>
<head><title>Example</title></head>

<body>
<table border="1">    (always a good idea to add a border when starting out.)
  <tr>
    <td>My </td>
    <td>blog.</td>
  </tr>
  <tr>
    <td>Left bar</td>
    <td>Main content</td>
  </tr>
</table>

</body>
</html>
```



# So what does that give us?



# Uh....

- Not really what I had in mind. But, that's all we told the browser to do -- we need to be a little more specific.
- Let's start by making this table take up the whole page.
  - Tables have the nice property of being able to specify their width and height easily in HTML.
  - Let's use "100%" for both -- that will make the table use up 100% of its parent element's size (here, the body tag).

```
<table width="100%" height="100%" border="1">
```

# Resizing cells individually

- Before we try it out again, let's also make it so that each cell uses up the right amount of space. We want our top row to be exactly 95 pixels high, but we want the bottom row to use up the rest.
  - This is much harder to do without tables -- if we were using a pure-CSS approach, we'd be doing things differently. But since we *are* using tables...
- Also, we want the left column to be 167px wide.

```
<html>
<head><title>Example</title></head>

<body>
<table width="100%" height="100%" border="1">
  <tr>
    <td width="167px">My </td>
    <td height="95px">blog.</td>
  </tr>
  <tr>
    <td width="167px">Left bar</td>
    <td>Main content</td>
  </tr>
</table>

</body>
</html>
```

Example



My

blog.

Left bar

Main content

# Looking better.

- There are still three problems we want to fix with the layout before moving on.
  - The space around the edge of the table... that'll be a problem. The client didn't want the blue title bar to have a margin around it.
  - The content is vertically-centered in each cell... we don't want that, either.
    - Easy: just add `valign="top"` to each cell.
  - We also need to merge the top two cells.
    - This one is easy, if a bit counter-intuitive: we remove the right cell, and add an attribute called `colspan` (column span) to the left cell with the value 2. That will make it act as if it were two whole cells.



# Let there be CSS

- The first problem, however, is pretty specific, and we'll need more than just HTML to deal with it
- We can apply a **style** to the document
  - The markup language that we define styles in is called CSS, for Cascading Style Sheets.
  - For now, we'll throw the CSS right into the tags -- this is called "inline" CSS. Usually, you abstract the CSS into a separate style sheet.
- We put this style information inside the **style** attribute.

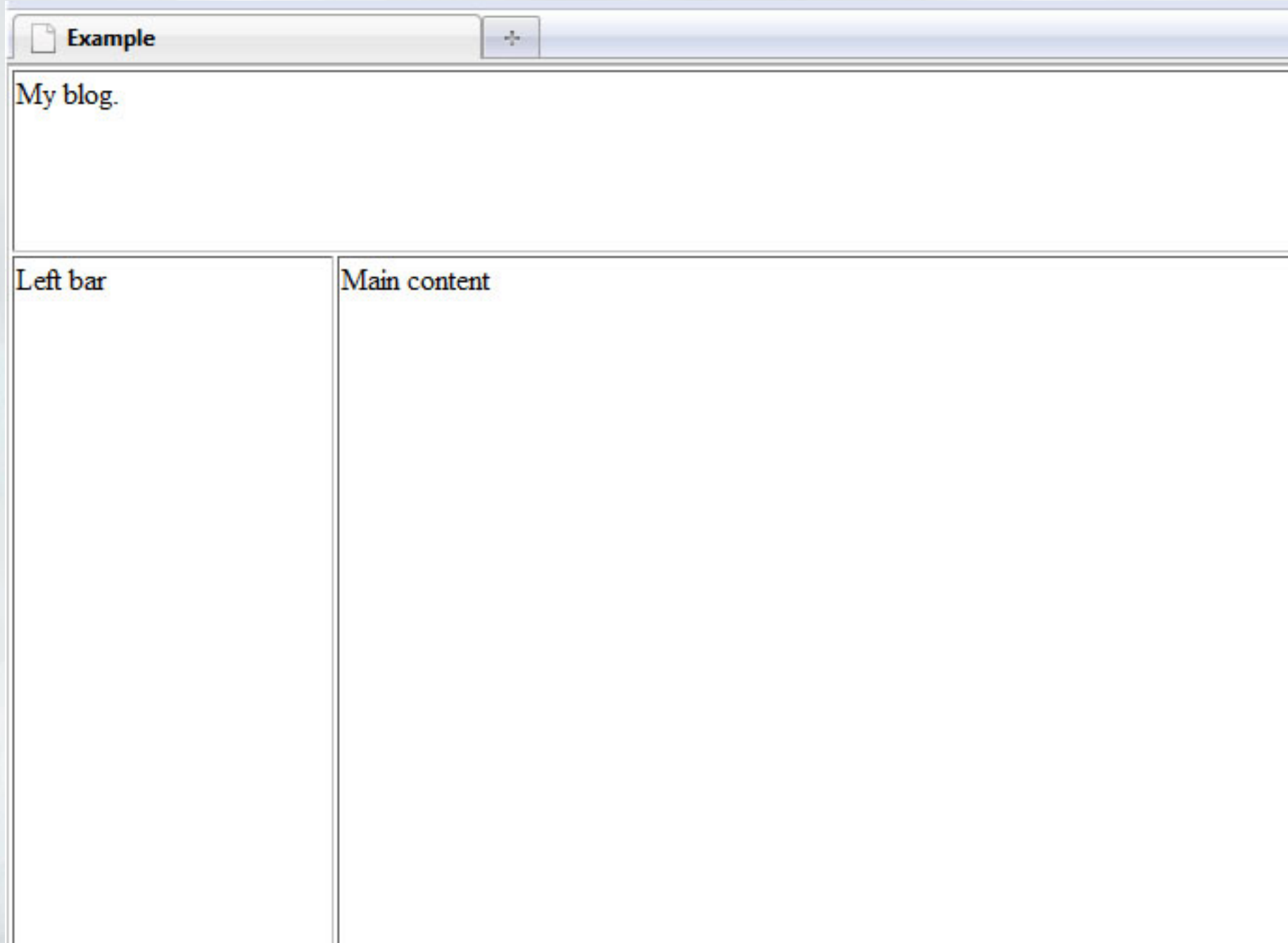
Note: Some problems can be solved with either HTML attributes or CSS styles; some can only be solved via one or the other.

# Adding some style to our page

```
<body style="margin:0px;">
```

- Note that the syntax used for specifying style attributes is somewhat like how HTML attributes work -- however, instead of `property="value"`, we use `property:value;` (with semicolons separating multiple styles).
- The property we're adjusting here is called **margin**. Margin is how much extra space should be around the given element
  - That is, how wide the element's "personal space" is.
  - Here, we don't want any.
- Why did I put this in the body tag, rather than table?
  - Experiment! Sometimes it's hard to tell what will work.
  - Remember to test on multiple browsers...

# Much better.



# Still pretty bad, though.

- Let's make this actually look like something. We want the top cell to have a blue background, right? So let's give it a style that will make it blue.
- The *background* property (in CSS) is a versatile way to specify backgrounds. Look it up for more details.
  - Here, we just want `background:#006699;`
- `#006699` is a **hexadecimal color code**. It's a compact way of specifying colors. I would recommend using a hex color-picker of some sort (google it, there are plenty) to get these.
  - Remember the `#` in the beginning!
  - You can use words for very common colors, like white, red, or black. E.g., `background:white;`

# The header text

- Let's also make the text inside the blue bar look like it should. First, we'll wrap it in an h1 element (header-one).
  - h1, h2, h3 ... h7 are various levels of header text. You should use them when appropriate (for accessibility).
- Then, we need to style that header element:
  - Need to make it white *color:white;*
  - Need to make it big *font-size:43px;*
  - Need to make it Verdana *font-family: Verdana;*

```
<h1 style="color:white;font-size:43px;font-family:Verdana;">My Blog</h1>
```

```
<html>
<head><title>Example</title></head>

<body style="margin:0px">

<table width="100%" height="100%" border="1">
  <tr>
    <td height="97px" valign="bottom" colspan="2"
      style="background:#006699;"
        <h1 style="color:white;font-size:43px;
          font-family:Verdana;">My Blog</h1>
    </td>
  </tr>
  <tr>
    <td width="167px" valign="top">Left bar</td>
    <td valign="top">Main content</td>
  </tr>
</table>

</body>
</html>
```

(Notice how we can split tags across multiple lines? HTML doesn't really care.)



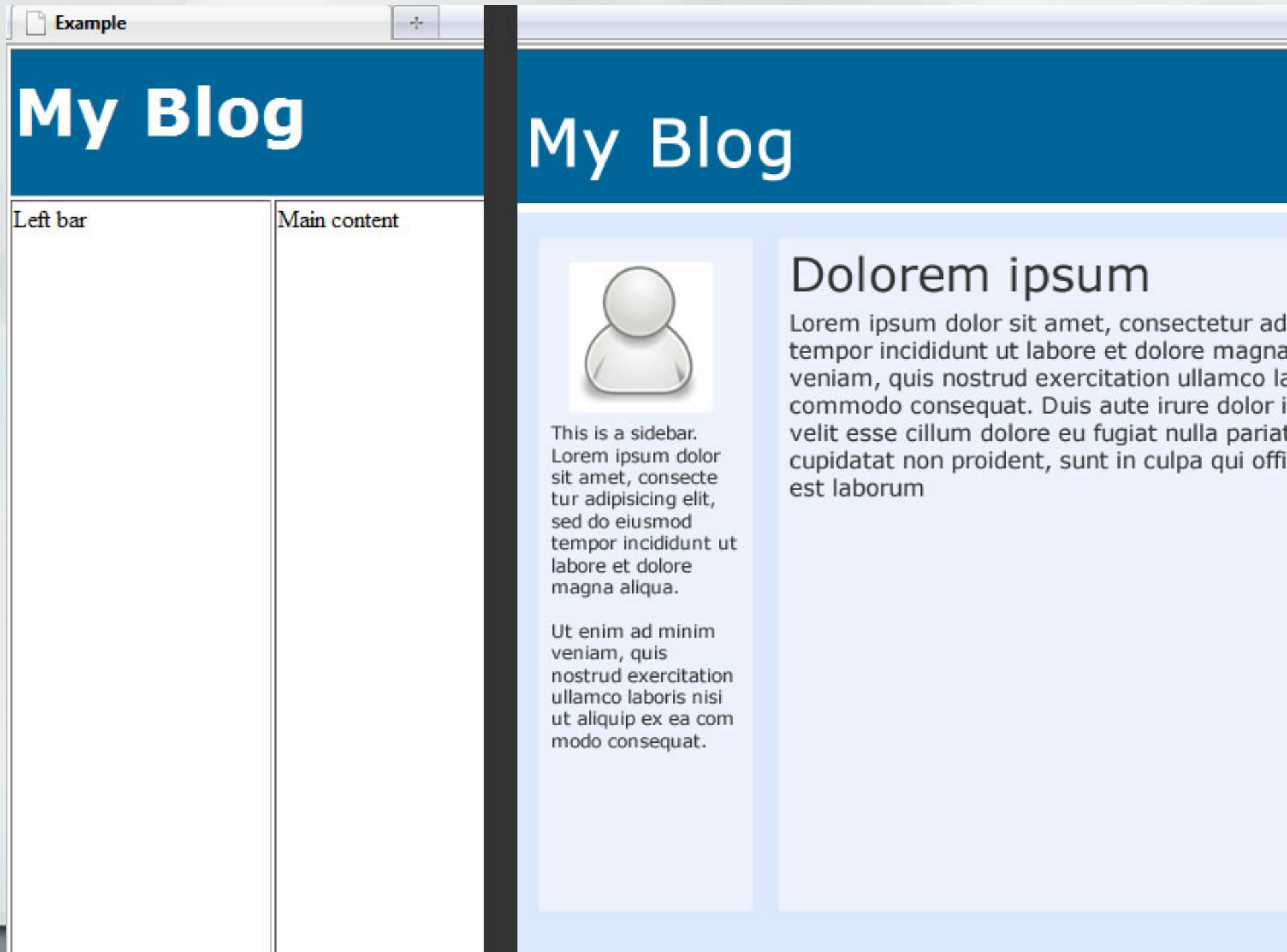


# My Blog

Left bar

Main content

# Better... but....



# What's going on?

- Unfortunately, the header tags all come bundled with their own (pretty crappy) styles
  - Lots of margin/padding and boldness
  - You can use e.g. Firebug to find out more
- We'll need to overwrite these.
  - `style="margin:5px;padding:0px;font-weight:normal; ...`

```
<h1 style="margin:5px;padding:0px;font-weight:normal;color:white;font-size:43px;font-family:Verdana;">My Blog</h1>
```

(there's a good reason styles are put in separate CSS files!)

# That fixes that, so let's move on.

- We need a border on the bottom of the header...
  - Add `style="border-bottom:7px solid white;"` to the cell
  - This gives the cell a solid (as opposed to dashed, for example) bottom border 7 pixels wide, colored white.
- Also, the page as a whole needs that light blue background.
  - Add `background:#DBE8FD;` to the body tag's style.
- And, let's get rid of that border on the table. Don't need it right now.



# My Blog

Left bar

Main content

# Not bad.

- That takes care of the header bar. Now, we need to add the off-white boxes for the actual page content.
- We *could* nest another table inside each cell, and style the cell of that table like we did before...
- But there's no need. We can use HTML's all-purpose "content box" element, the DIV.

```
<div style="">Anything you want</div>
```

Note that the DIV element has very few actual HTML attributes -- to really manipulate it, you need to use styles. And since all a DIV starts off as is an invisible box, you'll usually need to style it.



# Making the content DIV

- So, we want a box that fills up each table cell, but still has some margin around it.
- We can use the `margin` style to add in the spacing around the box, but how do we make each box fill up the entire cell?
  - This is actually a tricky problem, so we're just going to approximate a solution to save time.
  - We'll tell the left box exactly how wide to be, and for the right one, tell it 95%.
- Why can't we just tell each box 100% and then add margin?
  - Per the HTML standard, margin is added *on top* of width and height. So is padding from the containing element!
  - There are lots of little things like this (some browser-specific) that can make web design frustrating.



# My Blog

Left bar

Main content

# Looking better.

- One thing: the spacing between the two content areas is too wide. Why?
  - The margin from *both* boxes is applied, meaning there's 16px worth of margin between them.
- We want to set the left-margin of the right box (or the right-margin of the left box) to be zero pixels.
- To do this, we could either specify margins side-by-side, or override the margin-left property like so:

```
style="margin:8px;margin-left:0px;"
```

CSS has the property that later definitions will override previous ones. This comes in handy often.

# Finishing things up

- The text inside the content boxes has two problems:
  - Too close to the edge of the box.
  - Times New Roman? Not so much.
- We can fix both problems by styling the DIVs.
  - `font-family:Verdana;font-size:14px;padding:8px;`
- Also, we want a header in the right box -- since the header there is of secondary importance to the main title, we use h2.
  - `<h2 style="font-size:31px;font-family:Verdana;font-weight:normal;margin-top:0px;">Post Title</h2>`

# A bit more...

- And, we want to put an image in the left bar. To do so, we use the **img** tag:
  - ``
  - Note that this tag *self-closes*. Some tags do this, if they cannot contain any content.
- Also, we want that image centered. It's deprecated, but to keep things simple, let's just use the `<center></center>` tag.
  - `<center>anything</center>` will center that content.
- Finally, let's put a newline beneath it, to avoid cramping the text.
  - The `<br/>` tag is another self-closing tag. It stands for "break", as in, line break.



```

<html>
<head><title>Example</title></head>

<body style="margin:0px;background:#DBE8FD;">

<table width="100%" height="100%">
  <tr>
    <td height="97px" valign="bottom" colspan="2" style="background:#006699;border-bottom:7px
solid white;">
      <h1 style="margin:5px;padding:0px;font-weight:normal;color:white;font-size:43px;font-
family:Verdana;">My Blog</h1>
    </td>
  </tr>
  <tr>
    <td width="167px" valign="top">
      <div style="width:140px;margin:8px;background:#EDF3FE;height:90%;font-family:Verdana;
font-size:12px;padding:8px;">
        <center>
          
        </center>
        <br/>
        Left col.
      </div>
    </td>
    <td valign="top">
      <div style="width:95%;margin:8px;margin-left:0px;background:#EDF3FE;height:90%;font-
family:Verdana;font-size:14px;padding:8px;">
        <h2 style="font-size:31px;font-family:Verdana;font-weight:normal;margin-top:0px;"
>Dolorem Ipsum</h2>
        Right col.
      </div>
    </td>
  </tr>
</table>

</body>
</html>

```

# My Blog



Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

## Dolorem Ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

# Key ideas to take away

- Build your webpage step-by-step.
  - It's easier to tackle one thing at a time than try to write out your entire page at once.
- There are hundreds of HTML tags and CSS properties. Find a good reference and bookmark it.
  - For example, <http://www.w3schools.com>
- There are multiple ways to do *anything* in HTML/CSS.
  - There may not be a "best" way for something, but there usually are better and worse ways. Google your problem (i.e. "fixed-height two column CSS layout") if you need to.

# Can you modify the layout?

## My Blog



This is a sidebar.  
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

### Dolorem ipsum

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor [incididunt](#) ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

This is a sidebar.  
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.

Copyright 2010, Webmonkeys