net tuts+

presents

FROM PHOTOSHOP TO HTML

How to slice your designs like a pro

Jeffrey Way

ROCKABLE

Rockablepress.com
Envato.com

Foreword

I'm sure you've heard it before. "CSS is easy. Anyone can do it!" Granted, it may not be rocket science, but CSS is powerful enough to make you rip your hair out.

On a daily basis – if you listen closely – screams can be heard around the world from coders unsuccessfully attempting to force a design into position. It's not just a matter of learning the language; memorization is an easy task. The scream inducing points occur when you view your site in ten different browsers, all of which render your site at different levels of consistency. When such things occur, how do you fix them?

This is what separates the amateurs from the pros. By the end of this book, you'll be a member of the latter.

It won't be an easy task. As a matter of fact, I'm going to teach you to use techniques that seemingly make zero sense. Aren't you glad you purchased this book? All kidding aside, you're going to find that, just because it should work, doesn't mean it will. Just because your design looks great in Firefox, doesn't mean it will in Internet Explorer – in fact this is typically the case. To compensate, we'll be researching the unique "deficiencies" in each browser that will undoubtedly prove to give you trouble. When you finish the last page, you'll have the tools to battle the forces of evil and wrangle browsers – even those that are a decade old – to work for you, not against.

Jeffrey Way

AN INTRODUCTION

What This Book Includes

Packaged with this book, you will find:

1) A **directory of example files** made available for your reference. These files correspond to the example designs and HTML used in this book, and consists of: **Photoshop files for the Light Box Design and HTML / CSS files for the Light Box Theme**. These files and themes may be used freely in your projects, both commercial and non-commercial. However, they may not be redistributed or resold in any way. As you work through this book you may choose to either construct your own set of files from scratch, or use the example files as a guide.

2) A series of **six screencasts** covering the whole project from beginning to end.

What This Book Assumes

You should have at least an elementary understanding of HTML and CSS. You should be familiar with HTML tags and the most common CSS properties. For example, it will be assumed that you understand the syntax, and the relationship between the following HTML and CSS snippets:

HTML

```
  <div id="myDiv">
…some text goes here
  </div>
```

CSS

```
#myDiv {
   Width: 500px;
   Margin: auto;
   Background: red;
}
```

Additionally, you should have a basic understanding of Photoshop.

What This Book Does Not Assume

Nothing more will be assumed beyond a basic understanding of the syntax for these two languages.

Tools of the Trade

When converting a PSD to HTML and CSS, you'll be most efficient if you have the right tools at your disposal. Luckily, other than Photoshop, there are a plethora of programs and extensions available to you, most of them free. In this book, we'll be using the following:

Komodo Edit

https://www.activestate.com/komodo_edit/downloads/

There are a variety of free code editors for both the Mac and PC. A simple Google search will return several options. One that will beautifully serve users of both platforms is Komodo Edit.

Web Developer Extension for Firefox

https://addons.mozilla.org/en-US/firefox/addon/60

This nifty extension will allow you to edit your CSS inside of the browser, disable JS, apply rulers, validate the HTML and CSS, to name a few. It's an essential tool that nearly all developers have installed. You should as well.

IE Tester

http://www.my-debugbar.com/wiki/IETester/HomePage

Unfortunately, even after a decade, we must still compensate for older browsers – namely Internet Explorer 6 – when converting designs. Most likely, you'll have IE7 or IE8 installed though. In order to preview your designs in this browser, one option – if on the PC – is IE Tester. It allows you to view any site in IE5 – IE8.

BrowserShots.org

http://browsershots.org/

If you work on a Mac (or need to test in a larger variety of browsers), a second option is to use a site called Browsershots, to take a snapshot of your design in all of the browsers.

VMWare Fusion

http://www.vmware.com/products/fusion/

Especially in the design world, Mac users are quickly becoming the majority. Though Browsershots is definitely helpful, sometimes

the best option, when needing to test in IE6, is to use a program like VMWare Fusion. As they put it, “Windows is even better on a Mac!”. Once loaded, you can then easiy run IE Tester.

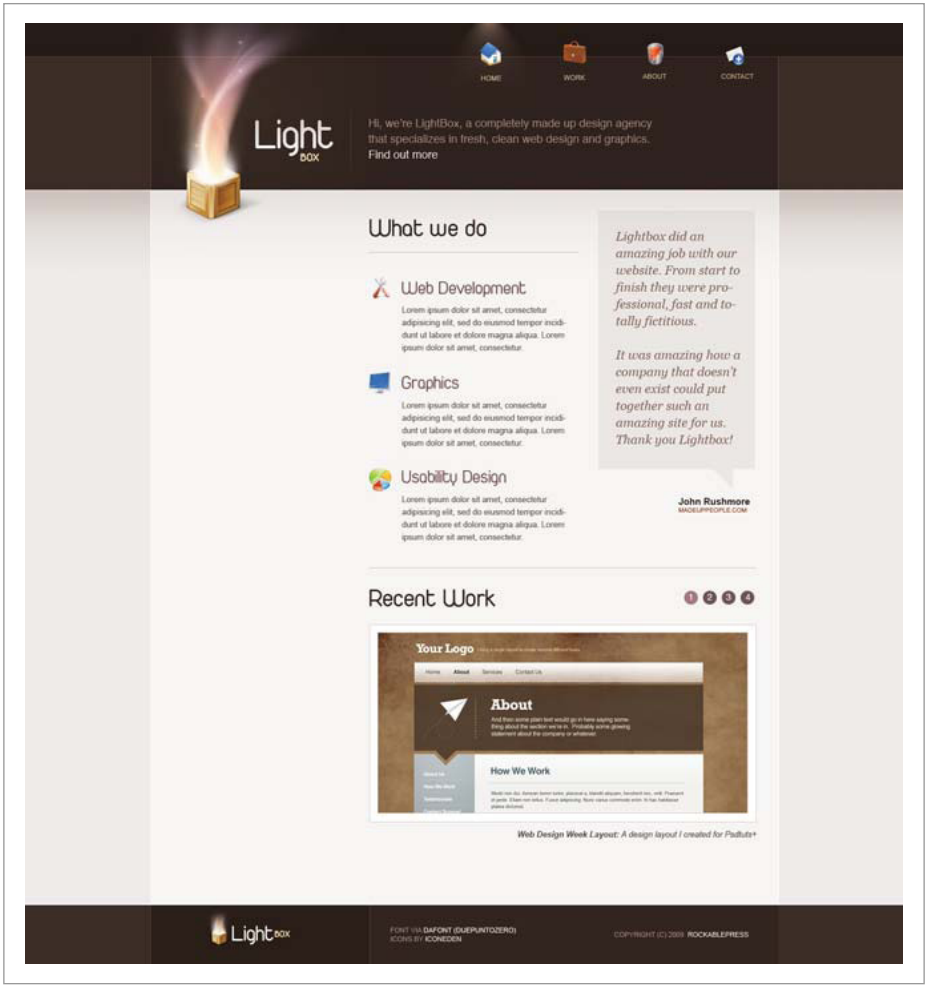
Coding The HTML

While it might seem natural to immediately begin working on the visuals of our website, this actually couldn't be further from the truth. Instead, we must first build our base, or the mark-up, and only once it's been completed can we move on to the visuals.

In this section, we'll analyze the design, and then translate the structure to HTML mark-up.

Coding The HTML

Consider the following layout:



We will be converting this exact design to a complete HTML / CSS website! When viewing such a design, we must begin to mentally format our layout from an HTML point of view. After a bit of observation, you might imagine the following layout:

You should begin thinking this way when viewing every design. This layout is essentially divided into three sections:

1. Header
2. Main Content
3. Footer

You'll find that most sites will have a similar structure. Some might have a few more columns, but in general, you'll always find these three components.

Creating the Basic Shell

Now that we've visualized our design from a coder's point of view, let's create the basic HTML formatting.

We begin by creating a "container" division. That way, if necessary, we can position our entire website simply by altering one div.

ROCK* TIP

Think of a div as a box. Within this div, we can insert a bunch of stuff (elements). Then, if we need to move all of that stuff around, we simply pick up our box and reposition it!

```
<div id="container">
...stuff
</div>
```

Next, add a few more divisions. Ignoring the specifics and focusing solely on the wrapping elements, let's add our three components:

```
<div id="container">
    <div id="header">
            header content goes here
    </div>

    <div id="main">
            main body goes here
    </div>
    <div id="footer">
            footer info goes here
    </div>
</div>
```

Referring back to our design, we could then begin inserting the correct elements: an image here, a blockquote there. If you've never converted a PSD before, you're already on your way! **It's as simple as visualizing the layout from a coder's perspective, and creating the markup to match it.**

The Markup

Let's now take the process a step further. Within our `#main div`, we'll need to insert additional wrapping divs for the "What We Do" section as well as the sidebar.

If we wanted to, we could simply name the two divs, "whatWeDo" and "right-sidebar." However, this method could potentially cause problems. What if, down the road, we instead decide to display a list of recent products? In that case, we'd have to rename this div, and all of its references within the stylesheet. Obviously, "whatWeDo" wouldn't make any sense!

What if we decide to switch the sidebar to the left side? As a general guideline, remember to name your elements according to what they are, and avoid presentational specifics. This is referred to as "semantics." For example, consider the `<i>` and `<em>` HTML tags. They essentially perform the exact same task: making text italic. With that said, their meanings are quite different. The `<i>` stands for "italic," while `<em>` stands for "emphasis." Notice the difference? One element refers to the styling (or presentation), while the other simply describes what text should be – emphasized.

ROCK* TIP

As a rule of thumb, if you have to edit both the HTML AND the CSS when adjusting the styling of your page, you're doing it wrong.

Always separate presentation from content.

Returning to our project, we'll simply use "primaryContent" and "secondaryContent."

```
<div id="container">
    <div id="header"></div>

    <div id="main">
        <div id="primaryContent"></div>
        <div id="secondaryContent"></div>
    </div><!-- end main-->
</div><!--end continer-->
<div id="footer"></div>
```

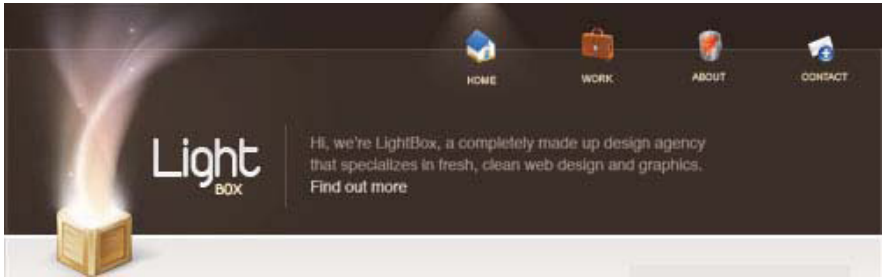
Also, take note of how our footer div is a sibling of our container div, not a child (an element that is nested inside of another). The reason is because we want the brown background to extend for the entire width of the user's browser window. This cannot be accomplished when nested inside a div with a stated width. Don't worry if this is confusing; we'll discuss it more in later sections.

ROCK* TIP

Do not touch your CSS file until the markup has been completed.

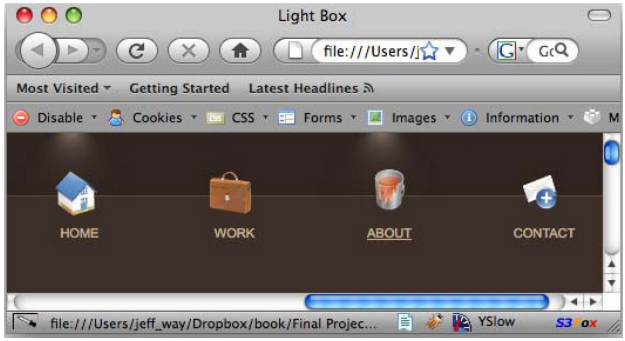
Header

Now that we have our basic structure, let's begin fleshing out each section. We'll start with the header.



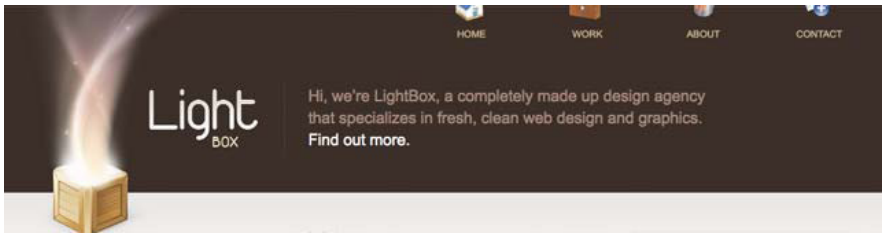
We immediately notice a "list" of navigation links in the upper right portion of the screen. Because it's a list, it seems appropriate to use an unordered list!

```
<div id="container">
<div id="header">
  <ul id="nav">
    <li id="li_home"><a href="index.html">Home</a></li>
    <li id="li_work"><a href="work.html">Work</a></li>
    <li id="li_about"><a href="about.html">About</a></li>
<li id="li_contact"><a href="contact.htm">Contact</a></li>
  </ul>
</div>
```



Notice how each list item has been given a unique id? This is an unfortunate necessity, due to the unique nature of our navigation. Return to the demo and hover over each link; you'll notice that a warm glow appears around each icon when you do so. To achieve this effect, we'll use background images. We need a way to target each list item in order to apply its respective background image – hence the unique ids.

Moving along, we now come to the logo and the slogan. We should wrap this section within its own div as well.



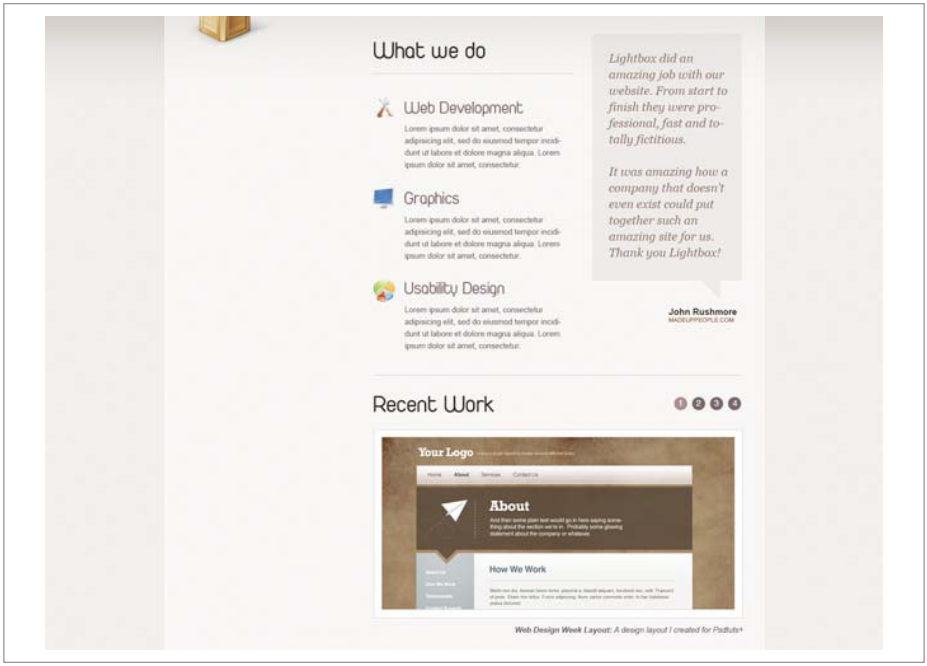

```
<div id="header">
    <ul id="nav">
      <li id="li_home" class="selected"><a href="index.
html">Home</a></li>
      <li id="li_work"><a href="work.html">Work</a></li>
      <li id="li_about"><a href="about.html">About</a></li>
      <li id="li_contact"><a href="contact.html">Contact</
a></li>
    </ul>

    <div id="slogan">
      <h1><a href="index.html">Light Box</a></h1>
      <img src="#" alt="Light Box" />
      <p>Hi, we're LightBox, a completely made up design
agency that specializes in fresh, clean web design
```

```
    and graphics.
          <a href="about.html">Find out more.</a></p>
        </div><!-- end slogan-->
    </div><!-- end header-->
```

We begin by placing the name of our site within a heading level one tag. It's considered a best practice for usability to make your logo link back to the home page. This can easily be implemented by wrapping the text within an anchor tag.

Next, we have that large glowing box graphic, and finally our slogan, which is appropriately wrapped in a paragraph tag. Don't worry about the actual image just yet, it will be available to us once we've sliced our PSD. For the moment, we're ONLY focusing on the markup.

Main Content



```
<div id="primaryContent">
    <h2 id="whatWeDo">What We Do</h2>
    <ul>
            <li id="first">
                    <h3>Web Development</h3>
                    <p> … generic text … </p>
            </li>

            <li id="second">
                    <h3>Graphics</h3>
                    <p> … generic text … </p>
            </li>

            <li id="third">
                    <h3>Usability Design</h3>
                    <p> … generic text … </p>
            </li>
    </ul>
</div><!-- end primaryContent-->
```

At this stage we wrap the "What We Do" heading within an h2 tag. Next, we have three different sections.

1. Web Development
2. Graphics
3. Usability Design

A beginner might assume that three divs would be appropriate here – however, that would be the incorrect choice.

ROCK* TIP

If someone asks you what this section is, you might respond by saying that it's a "list of services" offered by our fictional company. The keyword here is "list". As such, we should be using an unordered list.

Each list item contains an H3 tag and the generic “lorem” dummy text (represented by ... generic text ...). We’ll use the background property in our CSS file to add the icon and the unique font. This is the reasoning behind adding the “first”, “second”, and “third” ids to the list items. We need a way to latch on to each one individually from within our stylesheet.

SecondaryContent

Immediately after our primaryContent div, we need to add a blockquote element, which will display a random testimonial.

```
<div id="secondaryContent">
    <blockquote>
            <p>Lightbox did an amazing job with our website.
From start to finish they were professional, fast and
totally fictitious. </p>
            <p>It was amazing how a company that doesn't
even exist could put together such an amazing site for us.
Thank you Lightbox!</p>
    </blockquote>
    <cite title="MadeUpPeople.com">John Rushmore
<span>MADEUPPEOPLE.COM</span></cite>
    <div class="arrow"></div>
</div><!-- end secondaryContent-->
```

One might think that we can simply remove the div entirely and float the blockquote to the right. At all times, we should be evaluating our code. Technically, that would be acceptable; however, we must always consider the future. What if, a few months down the road, we decide that the site needs a new page with a two-column layout? We'd unfortunately have to return to the HTML file to revise the markup. To prevent this, we should go ahead and create the second column. Then, we can use a "body" trick to determine whether each new page should consist of one column or two. You'll learn this trick soon!

There are two things worth noting in this block of code:

1. The cite tag might seem foreign to you. Luckily, it's quite simple. Cite tags are used exactly how you would imagine – to indicate a citation. Yes, we can get away with using something like a span tag, however, if a tag was created specifically for such a purpose, it's more semantic to use it.

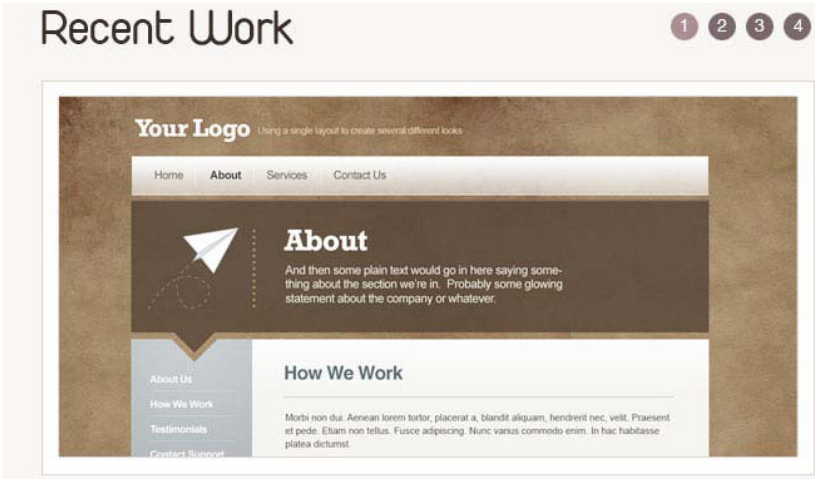
2. That final “arrow” div is not semantic. It’s required in order to create the tiny arrow at the bottom of the blockquote. **Did you know that the illusion of shapes can be created with CSS?** The alternative to this method is to use a background image and position it in the proper location; but that’s not any fun! We’ll review the basic technique shortly, but, if you’d like to learn more, be sure to watch my video tutorial on Nettuts+. http://net.tutsplus.com/videos/screencasts/fun-with-css-shapes/.

Recent Work

As with any portfolio, we need to create a fluid way to offer samples of our work. Right now, we’ll create a basic shell, which will consist of:

1. A list of items to browse.
2. An image of each website.
3. A brief description of each job.

Because we don’t want to overflow the page with images, we’ll ultimately use JavaScript to create the transition effect. When a user clicks on one of the numbered items, the image will fade out, switch to the selected one, and fade back in. You’ll learn how to do this in the ‘Bonus Chapter’ - as this book’s primary goal is to teach CSS and slicing techniques.



Web Design Week Layout: *A design layout I created for Psdtuts+*

```
<div id="recentWork">
    <h2>Recent Work</h2>
    <ol>
            <li class="chosen"><a href="images/recentWork/
recentWork1.jpg" title="A design layout I created for
Psdtuts+">1</a></li>
            <li><a href="images/recentWork/recentWork2.jpg"
title="A pastel layout I created for Psdtuts+">2</a></li>
            <li><a href="images/recentWork/recentWork3.jpg"
title="A grass layout I created for Psdtuts+">3</a></li>
            <li><a href="images/recentWork/recentWork4.jpg"
title="A dark layout I created for Psdtuts+">4</a></li>
    </ol>
    <div class="image_display">
            <img src="#" alt="Recent Work" />
    </div>
    <p><strong>Web Design Week Layout:</strong> A design
layout I created for Psdtuts+</p>
</div><!-- end recent work-->
```

This “Recent Work” div requires a bit more work. It actually takes up the entire width of the main content. Considering this, it can’t be placed within either the primary or secondary content divs.

1. Per usual, we start with our H2 tag.

2. Next, we use an ordered list to create the circular buttons. Within this list, we add a nested anchor tag, which links to the next image.

3. We insert our main image. At first glance, it might seem unnecessary to wrap the image within this “image_ display” div. It’s actually a necessity. If you’ll look at the image closely, you might notice how it has a 1px border, some padding, and then another 1px border around the padding. Unfortunately, we can’t achieve this effect without creating a wrapper div. It’s a shame, but sometimes sacrifices must be made for aesthetics.

4. Finally, after the div, we add a simple paragraph tag that describes the image above.

The Footer

Earlier in this chapter, I mentioned that we want the brown background to extend for the entire width of the user’s screen. To achieve this effect, we’ll have to use a bit of trickery.

```
<div id="footerWrap">
    <div id="footer">
            <div id="footerLogo">
                    <a href=""><img src="#" alt="Light Box"
/></a>
            </div>
                    Font via <strong><a href="http://www.
dafont.com/duepuntozero.font">dafont (Duepuntozero)</a>
<strong><br />
                    ICONS by <strong><a href="http://
iconeden.com/icon/bright-free-stock-iconset.html">
iconeden</a></strong>
                    <p id="footerInfo">&copy; 2009  <a
href="http://rockablepress.com/">rockablepress</a></p>
            </div><!-- end footer -->
</div><!-- end footerWrap-->
```

We've added an additional wrapping div called "footerWrap". This div will ultimately have its width set to 100% (or the width of the browser window). Add in a repeating background image, and we have our tile effect. However, the text within the footer still needs to line up with our container div. This is why we're keeping the `#footer div`. Just as with the container, we can set a specific width and center it on the page. We'll do this in the CSS chapter.

Inside our footer, we've placed an image containing the Light Box logo as well as a few links crediting the creators of the icons and the heading font.

Markup Complete!

That's it for the home page! It wasn't too difficult! If you're working along – and I hope you are – your final code should look like this:

```
<div id="container">
    <div id="header">
            <ul id="nav">
                    <li id="li_home" class="selected"><a
href="index.html">Home</a></li>
                    <li id="li_work"><a href="work.
html">Work</a></li>
                    <li id="li_about"><a href="about.
html">About</a></li>
                    <li id="li_contact"><a href="contact.
html">Contact</a></li>
            </ul>

            <div id="slogan">
                    <h1><a href="index.html">Light Box</a>
<h1>
                    <img src="#" alt="Light Box" />
                    <p>Hi, we're LightBox, a completely
made up design agency that specializes in fresh, clean web
design and graphics. <a href="about.html">Find out more.
<a></p>
            </div><!-- end slogan-->
    </div><!-- end header-->

    <div id="main">

            <div id="primaryContent">
                    <h2 id="whatWeDo">What We Do</h2>
                    <ul>
                            <li id="first">
                                    <h3>Web Development</h3>
                                    <p> … filler text … </p>
                            </li>

                            <li id="second">
                                    <h3>Graphics</h3>
```

```
                                        <p> … filler text … </p>
                                </li>

                                <li id="third">
                                        <h3>Usability Design</h3>
                                        <p> … filler text ...</p>
                                </li>
                        </ul>
                </div><!-- end primaryContent-->

                <div id="secondaryContent">
                        <blockquote>
                                <p>Lightbox did an amazing
    job with our website. From start to finish they were
    professional, fast and totally fictitious. </p>
                                <p>It was amazing how a company
    that doesn't even exist could put together such an amazing
    site for us. Thank you Lightbox!</p>
                        </blockquote>
                        <cite title="MadeUpPeople.com">John
    Rushmore <span>MADEUPPEOPLE.COM</span></cite>
                        <div class="arrow"></div>
                </div><!-- end secondaryContent-->

                <div id="recentWork">
                        <h2>Recent Work</h2>
                        <ol>
                                <li class="chosen"><a
    href="images/recentWork/recentWork1.jpg" title="A design
    layout I created for Psdtuts+">1</a></li>
                                <li><a href="images/recentWork/
    recentWork2.jpg" title="A pastel layout I created for
    Psdtuts+">2</a></li>
                                <li><a href="images/recentWork/
    recentWork3.jpg" title="A grass layout I created for
    Psdtuts+">3</a></li>
```

```
                        <li><a href="images/recentWork/
recentWork4.jpg" title="A dark layout I created for
Psdtuts+">4</a></li>
                    </ol>

                    <div class="image_display">
                        <img src="#" alt="Recent Work" />
                    </div>
                    <p>
                         <strong>Web Design Week
Layout:</strong>
                         A design layout I created for
Psdtuts+
                     </p>
            </div><!-- end recent work-->
    </div><!-- end main-->
</div><!--end container-->

<div id="footerWrap"><!-- needed to extend the brown
background for the entire screen -->
    <div id="footer">
            <div id="footerLogo">
                    <a href=""><img src="#" alt="Light Box"/>
</a>
            </div>
                    Font via <strong><a href="http://www
dafont.com/duepuntozero.font">dafont (Duepuntozero)</a>
<strong><br />
                    ICONS by <strong><a href="http://
iconeden.com/icon/bright-free-stock-iconset.
html">iconeden</a></strong>
                    <p id="footerInfo">&copy; 2009  <a
href="http://rockablepress.com/">rockablepress</a></p>
            </div><!-- end footer -->
</div><!-- end footerWrap-->
```

2

Slicing the PSD

We're finally ready to begin slicing our PSD. Take a moment to look over the design in Photoshop. Try to pinpoint each image that will be needed. Remember, many features can be recreated with CSS; so be creative.

The Plan

As I see it, we'll need the following items:

1. Navigation icons
2. Logo

3. Light box image
4. All of the headings and their respective icons
5. “Recent Work” snapshot”
6. Smaller logo within the footer
7. Background gradients

Other than the seven items listed above, everything else can be recreated with simple CSS!

Photoshop

Let’s visit Photoshop and begin! When it comes to “slicing” PSDs, there are two acceptable practices:

1. The first group believes that you should use Photoshop’s “slice” tool to get the job done. It was created specifically for this function, so why not use it?
2. The second prefers to use a mixture of cropping, and copy/paste methods, because they feel that it makes for a quicker process.

Both methods are perfectly valid. I recommend that you try both, and choose which ever proves to be

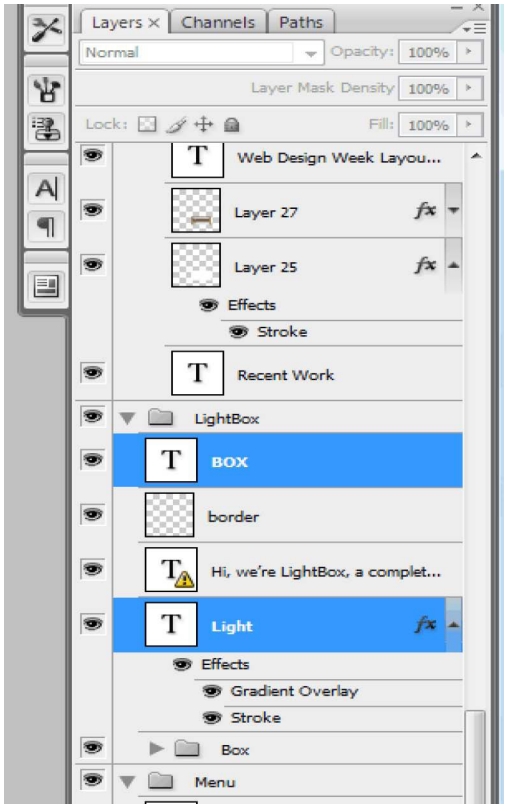
ROCK* TIP

An easy way to target a specific layer is to turn on “Auto Select.”

a) Click on the Move Tool (V).

b) Make sure that the “Auto Select” checkbox is checked. It’ll be in the upper-left side of the screen.

c) Click on the logo. Notice how it immediately selected the appropriate layer?



fastest... for you. I typically use the second method. As such, that's what you'll be using in this book!

We'll begin by grabbing the logo. Within the layers palette, open the "Lightbox" folder and search for the "Light" and "Box" layers.

Okay, now make sure that you're working along for this next part. It's the only way that you'll remember the following steps.

1. You'll see that the logo is split into two layers. Click in the small square icon within the first layer. You'll immediately see the marching ants.

2. While still holding **Command/Ctrl**, press and hold the Shift key, and click inside of the square icon of the next layer.

3. While still holding **Command/Ctrl** and **Shift**, press the letter "**C**" to copy the layers.

4. Let go of **Shift**, and now press "**N**." This will create a new document.

5. A new document dialog will appear, with the width and height values specifically set to the dimensions of your selection. Press **Enter**, and finally **Command/Ctrl V**, to paste the contents into the new document. Ta da!

The "Save for Web" Pattern

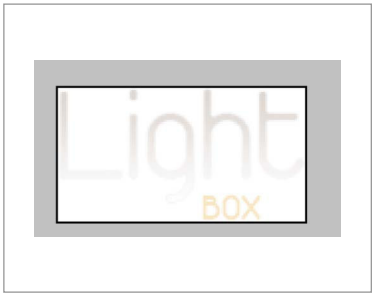
If you're somewhat new to Photoshop, all of these keyboard shortcuts might be a bit daunting. We could just as easily have utilized the toolbar locations for each command. However, this way is better, and is how the pros work!

Now is the perfect time to take a moment and memorize this pattern. You will use it many times. From this page on, this series of commands will be referred to as "The Pattern":

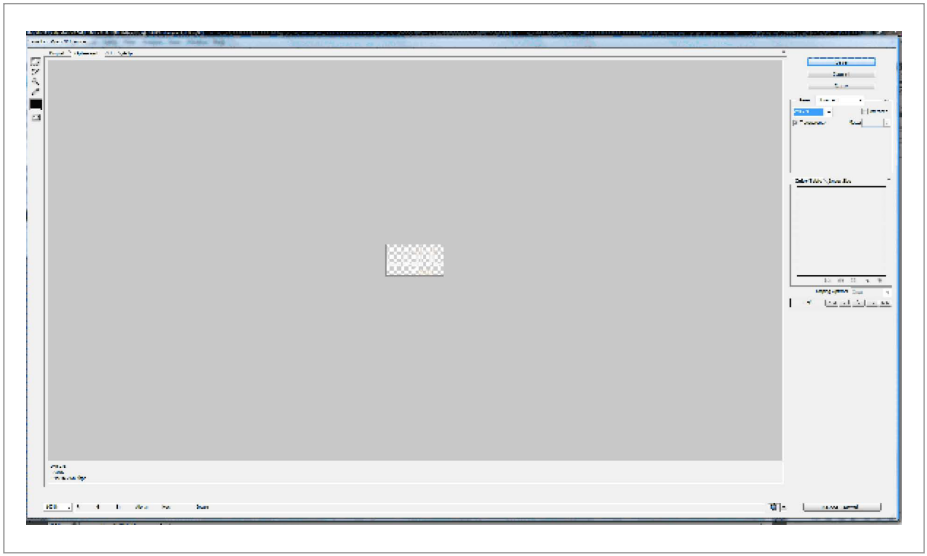
1. Find your desired layers.
2. Press and hold **Control** or **Command**
3. Click inside of the square icon of the layer.
4. If you need to copy multiple layers, hold **Shift** and click in the square icon of the next layer. Repeat as necessary.
5. Press **Comm/Ctrl + Shift + C** to copy the layer(s).
6. Press **Comm/Ctrl N** to create a new document.
7. Press **Comm/Ctrl V** to paste in the new layer.

Practice this several times until the pattern has been committed to memory. Once you've done so, continue on.

Returning to our project, you should now have a new document containing the logo on top of a white backgrund.



We obviously don't want that ugly white background; so click on the eyeball next to the "background" layer to deselect it. To save this image, go to *File > Save for Web and Devices* – or be a pro and press *Control + Shift + Alt + S* (or *Command + Shift + Option + S* for Mac users).

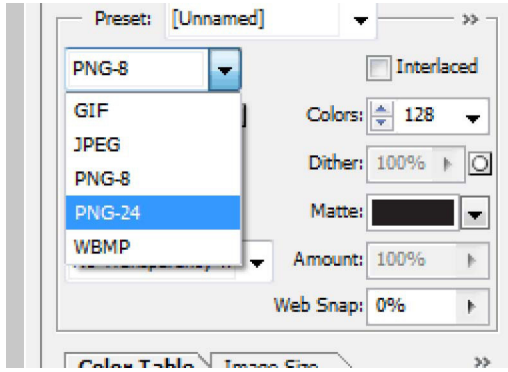


In order to save an image with partial or full transparency, we have two options: GIF and PNG. Years ago, GIF was the standard option. However, in recent years, PNG has proven to be the smartest decision for a couple of reasons:

- Compression is far more efficient than GIFs.
- The PNG-24 format offers variable (alpha) transparency, rather than 100% transparent or opaque pixels.

In this case, we want to make sure that the edges around our logo are perfect. PNG-24 is the correct choice. If, on the other hand, you don't need to render transparency, PNG-8 would be the smarter (and lighter) choice.

Best Practices for Image Types



1. **JPEG** – Photos and images with gradients.

2. **GIF** – Only needed when image is animated.

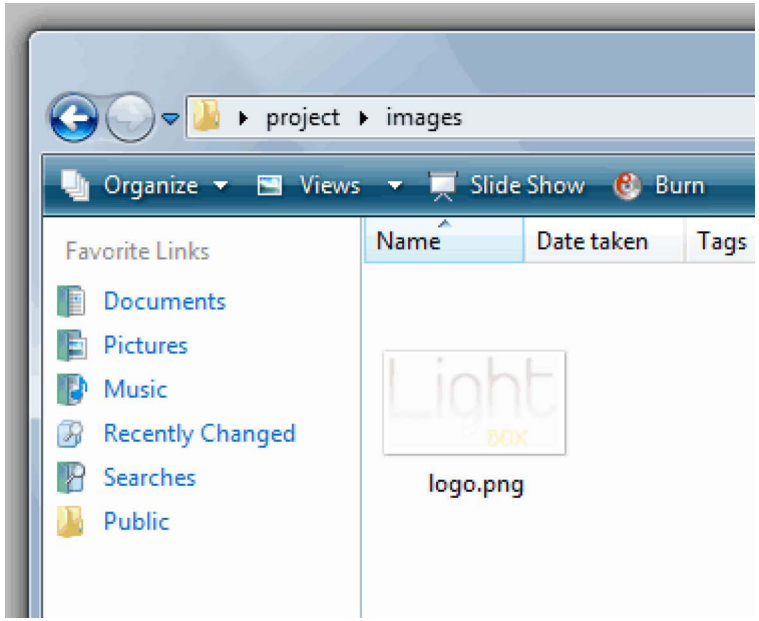
3. **PNG-8** – The best choice for graphics without transparency.

ROCK* TIP

If an image requires a transparent background, always use the PNG-24 format.

4. **PNG-24** – Larger file size, but renders transparency and image quality perfectly. Transparency not supported in IE6 and below.

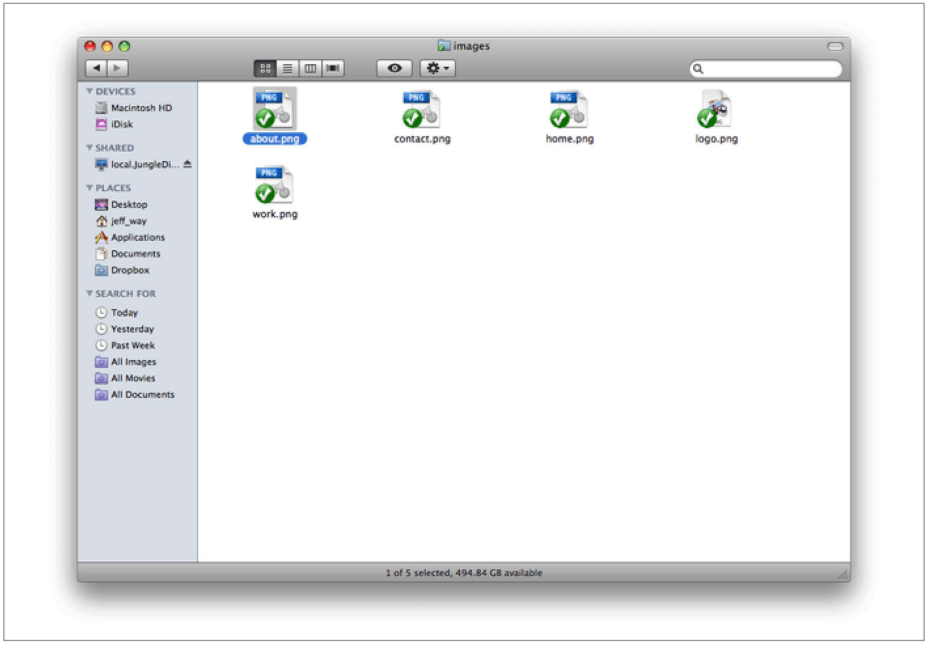
Name the image “logo.png”, and save it to the “images” folder of your project. Congratulations; one down!



Next, let's grab all of our navigation icons.

Once again, with "**Auto Select**" enabled, click on the tiny "house" icon, and then follow the exact same steps as listed in "The Pattern" (page 36). It's as simple as that. Because you took a few moments to memorize these keyboard strokes, you'll find yourself slicing and dicing in record time!

Using this same technique, save the remaining navigation icons to your images folder.



CSS Sprites

Though not entirely necessary, it's considered a best practice to use CSS "sprites" when possible. Don't worry if you don't know what these are just yet. While we'll go through them in more detail later, here is a quick definition:

> *Sprites allow us to group multiple images into one. Then, when the browser loads the page, it only needs to load one large image, rather than several smaller ones. Fewer HTTP requests equal faster page loads!*

With a sprite image, we can use background positioning in conjunction with the width/height properties to "frame" only our desired piece of the image. Ultimately, this means that our page loads faster! Now, let's move on to the "What We Do" section of the home page.

What We Do

1. Click on the "Web Development" heading. In this case, we need to grab both the title and the icon to the left. To do so, once again hold down Command / Control and click on the layer to select the text.

2. Next, hold down the “Shift” key and click on the icon layer as well (just below the text layer). Now, both layers will be selected simultaneously.

3. Repeat “The Pattern” (page 36) to create a new document and paste in the heading.

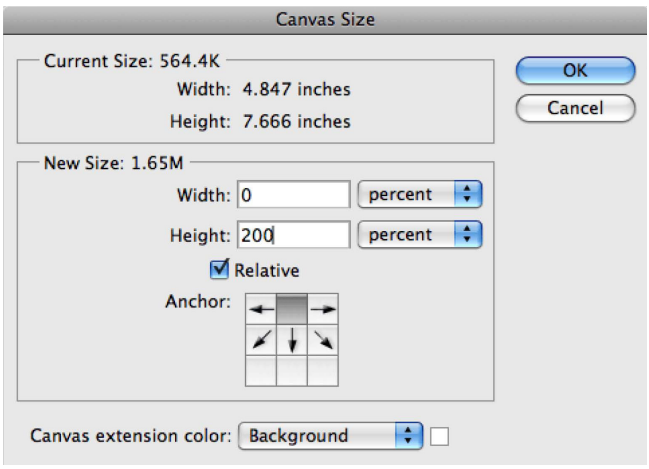
4. Before you save, we need to add the other two headings as well. That’s what makes it a sprite.

5. In the menu bar, go to *Image* > *Canvas Size*.

6. Check “Relative” and change the unit of measurement to “Percent.”

7. Set the anchor to the top-middle.

8. Within the “height” textbox, type 200. This will stretch our canvas so that it’s two times as large as its current form. This will allow us to conveniently place two more headings.



9. Return to your original PSD tab, and perform the same steps for the “Graphics” heading and icon. However, this time, rather than creating a new document, simply copy them and then return to your sprites PSD and paste. Repeat this process one last time for the “Usability Design” heading and icon.

When finished, you should end up with this:

Don’t forget to position your headings appropriately so that each one is properly underneath the former. Use the arrow keys to position each image accordingly. When finished, save for the web and place it in your images folder. Well done – your first sprite!

Footer

Next, let’s grab the logo from the footer.

You should be a pro at this by now. Just auto-select the layer, and follow "The Pattern." Simple!

Backgrounds

Now that we've gathered most of our images, we only need to grab our backgrounds, and we'll be finished.

1. Select the marquee tool, or press "M".

2. With all the layers turned on, go to *Layer* > *Flatten Image*.

3. Select a long strip of the background. It only needs to be a few pixels wide or so. We'll use CSS to repeat this pattern for the entire width of the window. Once you've done so, once again follow "The Pattern" (page 36) to save for the web.

4. The only other background we need is the one from the "container" section. It may look like a solid color, but it's actually made up of subtle gradients. Remember – subtlety is key here! As before, grab a thin and long strip and follow "The Pattern."

That's It for Now

Believe it or not, we've just finished the bulk of our slicing. Though we'll need to return for a few loose ends, the rest of the job can be done with CSS!

Revising our HTML

Now that we have all of these images at our disposal, we need to update our index.html document accordingly. Mostly, we'll be using the "background" CSS property to place images, however we do need to insert two image tags directly into our markup.

Find the div with an id of "slogan". This section needs to be updated in order to include the image of the Light Box.

```
<div id="slogan">
    <h1><a href="index.html">Light Box</a></h1>
    <img src="images/box.png" alt="Light Box" />
    <p>Hi, we're LightBox, a completely made up design
agency that specializes in fresh, clean web design and
graphics. <a href="about.html">Find out more.</a></p>
</div><!-- end slogan-->
```

Next, we must include the "Recent Work" image. Append the "image_display" div with our newly sliced image.

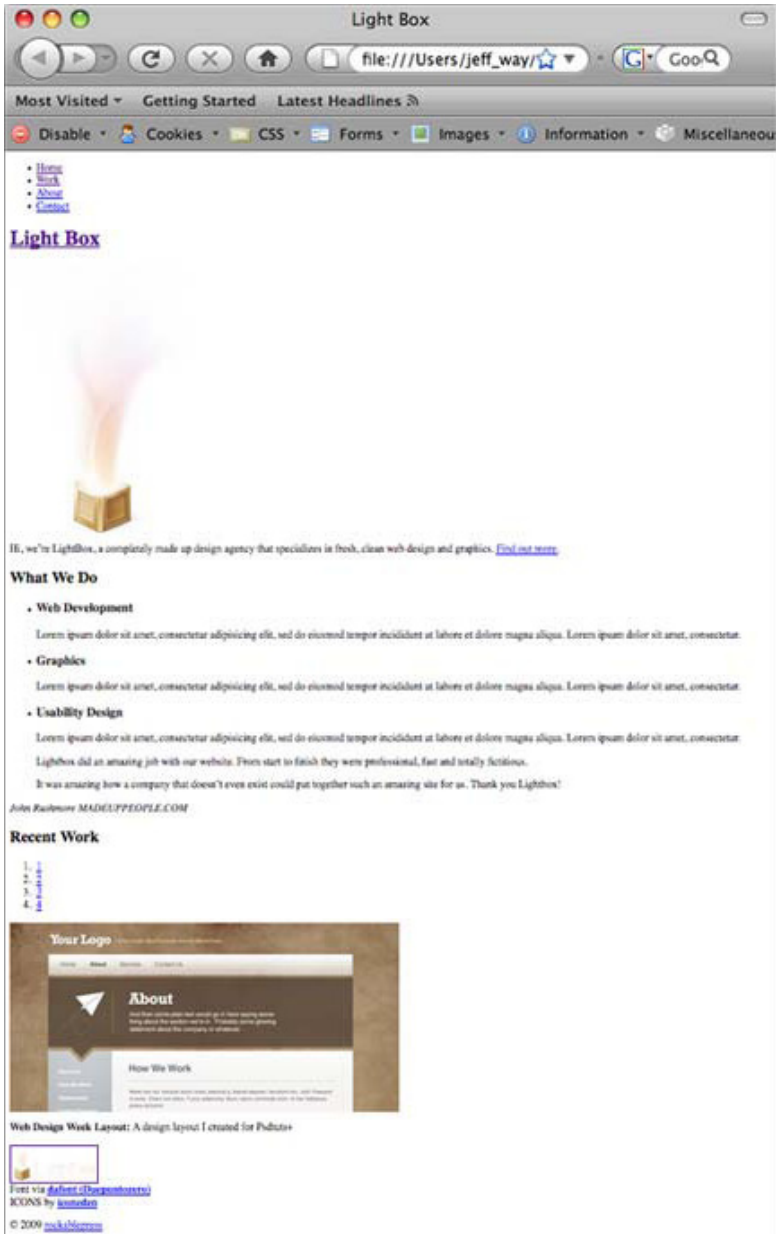
```
<div class="image_display"><!-- needed to apply a border to
both the image and the edges around the padding-->
    <img src="images/recentWork/recentWork1.jpg"
alt="Recent Work" />
</div>
```

Lastly, we must add the source for the logo within the footer.

```
<div id="footerLogo">
    <a href=""><img src="images/footerLogo.png" alt="Light
Box" /></a>
</div>
```

With that done, we've completed the mark-up for our home page!

View it in the browser, and you'll see:



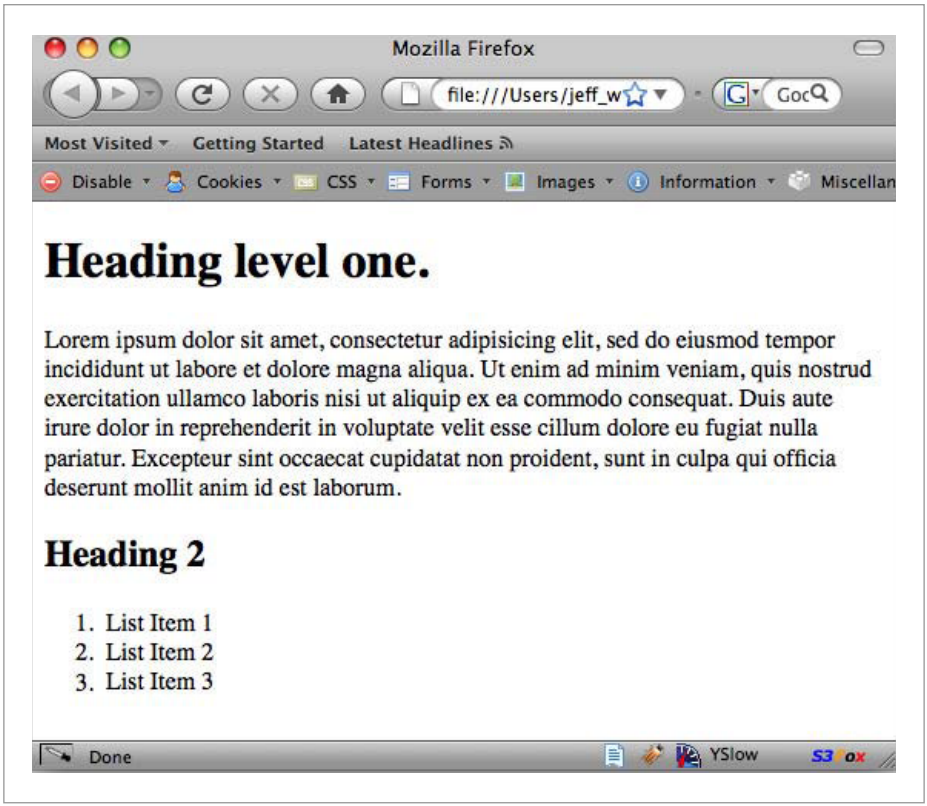
3

Coding the CSS

Every developer has his or her own preferences when coding a website. For example, many prefer to use a CSS reset stylesheet, while others find it to be a waste of time. Yours truly personally prefers the former. As such, you'll be doing so as well – at least in this book.

CSS Resets

Create a new HTML document and paste in any code you like. Add a few paragraphs, maybe a heading or two – anything you want. Next, view this page in your bowser. Here's what I ended up with:



Knowing that we haven't added an ounce of CSS:

1. Have you ever wondered how the heading tags are bold?

2. What about the slight margins around the paragraphs?

3. What about that subtle spacing around the browser window? Where does that come from? Shouldn't the text be right against the edge?

Each browser utilizes a bit of "default CSS" to automatically style your markup. At first glance, you might think that this is extremely helpful. Many professionals would argue this very thing, in fact. However, I, and many others, tend to feel that these default stylings do more harm than good. If every browser implemented them identically, this wouldn't be a problem. Sadly this is not the case.

How can we create a website that looks the same in all browsers if IE and Firefox interpret list items differently?

What's the Solution?

To work around this issue, which is sure to bring you massive headaches when first getting started, we'll create our own "reset" stylesheet, which will even the playing field for all browsers. That way, we have 100% control.

ROCK* TIP

If a "reset" sounds complicated, don't worry. It's quite simple. Mostly, it involves zeroing out the margins and padding.

Create a new CSS file called "reset.css" and save it in a new CSS folder within your project root. Inside this file, begin by adding the bare essentials.

```
Body,h1,h2,h3,h4,h5,p,ul,ol,li,blockquote,form,input {
    margin: 0; padding: 0;
}
```

Save your file, import it into your test document, and reload your browser. Notice the difference? Now, no formatting has been

applied. Experiment by viewing this page in every browser on your computer – it looks the same, right? That's the idea!

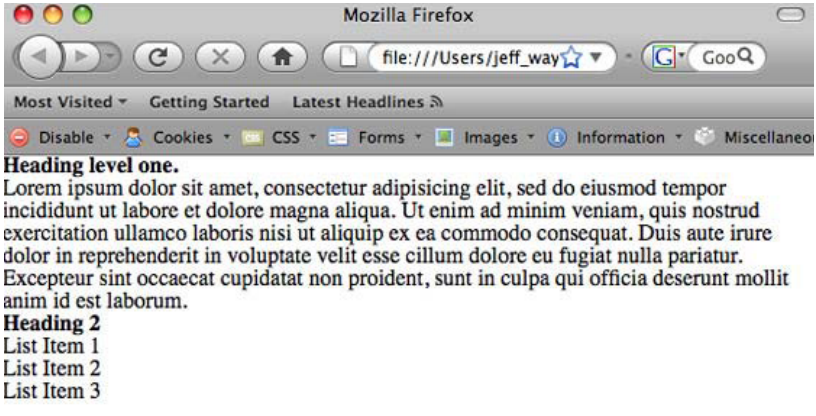
While this will definitely help, we need to make our "reset.css" stylesheet more robust. Erase what you currently have, and paste the following code in its place:

```
html, body, div, span, object, iframe, h1, h2, h3, h4, h5,
h6, p, blockquote, pre, a, code, del, em, font, img, small,
strike, strong, dl, dt, dd, ol, ul, li, fieldset, form,
label, table, caption, tbody, tfoot, thead, tr, th, td {
    margin: 0;
    padding: 0;
    border: 0;
    outline: 0;
    font-size: 100%;
    vertical-align: baseline;
    background: transparent;
}
```

```
body {
    line-height: 1;
}
ol, ul {
    list-style: none;
}
blockquote, q {
    quotes: none;
}
blockquote:before, blockquote:after, q:before, q:after {
    content: '';
    content: none;
}
```

You don't need to worry about this too much. Just rest assured that this code removes all browser styling. That way, you have 100% authority over your code – as you should! In addition to zeroing out the margins and padding, we also set the font-size of all elements to 100%, remove any outlines that Firefox might add, remove the bullet points from our list items, and so on.

As we'll be using this file in all of our projects, be sure to make a copy and store it in a "web dev folder", or something similar for later use.



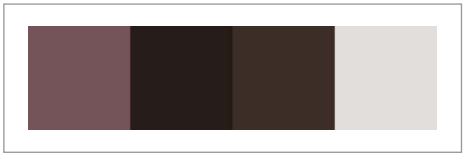
DEFAULT.CSS

Now, we need to create our workhorse stylesheet. Create a new file in your text editor and save it as “default.css” within your CSS folder. Paste in the following:

```
/*
Theme Name: Light Box
Description: Training for Rockable Press.
Slicer: Jeffrey Way
Designer: Collis Ta'eed


Colors:

text:              #745459
dark brown:        #251b17
light brown:       #3b2c25
white:             #e1dedb

*/

/* Generic Styles */

body {
    background: #f1efee url(images/bg.png) repeat-x;
    text-align: center;
    font-family: helvetica, arial, sans-serif;
    line-height: 25px;
}
```

We should always include a brief overview of our theme. In this case I’ve added four pieces: “Theme Name, Description, Designer, and Slicer”. Technically, this section isn’t required, but it’s generally considered to be good-practice to include it.

Remember Your Colors



I've found it helpful to hard code my primary colors into my stylesheets as a comment. That way, if I forget which hex value I need, I can scroll to the top of my file to retrieve the appropriate value, rather than returning to Photoshop and using the eyedropper tool.

Body

Typically, the "body" tag will be the first element you style.

1. First, we set the background property equal to the bg.png image that we sliced. Remember – if we don't also include a background color, our website will look very strange once the image has ended. To create a smooth transition, we set the background color equal to the color of our bg.jpg image at the very bottom – in my case, `#f1efee`. Yours might differ slightly.

2. Next, we align the text to the center. This has become a habit of mine that could truthfully be removed at this point. Originally, it was used as a hack to center a website in Internet Explorer 5. By aligning the body to the center, it will incorrectly center everything, rather than just the text in this browser. It's not 100% necessary anymore, but I still typically include it – if only out of habit.

3. Finally, we designate our font as "Helvetica" (as well as a few alternatives) and set a nice line-height, which will provide our text with some breathing room. Why Helvetica? Because that's the font that was used in our PSD, of course.

General Styles

Because we've reset every element, we should set our own values for a few common elements. By returning to Photoshop and checking the text size, color, and line-height, we can then use those exact values in our styleseet. Simply go to *Window* > *Character* in Photoshop.


```
h1, h2, h3, h4, h5 {
    margin: 25px 0;
}

h2 {
    font-size: 41px;
    color: #2f231e;
    padding-bottom: 25px;
    margin-top: 45px;
    border-bottom: 1px solid #cec5c0;
}

h3 {
```

```
    color: #745459;
    font-size: 30px;
    line-height: 33px;
}

.single h3 {
    font-size: 24px;
}

h4 {
    color: #2f231e;
    line-height: 20px;
    font-size: 16px;
    text-transform: uppercase;
    }

p {
    color: #645a56;
    font-size: 13px;
    font-family: arial;
}

a {
    text-decoration: none;
    color: black;
}

a:hover {
    text-decoration: underline;
}
```

These values were taken directly from their counterparts within the PSD. To verify, with the Character window open, click on a few headings and compare.

Container

With our general styles out of the way, we now need to work on our "container" div. This division will be the wrapper for our entire website.

```
#container {
    width: 1002px;
    margin: 0px auto;
    text-align: left;
    position: relative;
    background: #f7f5f4 url(../images/containerBG.png)
repeat-x;
}
```

If you return to your PSD, grab the ruler tool, and measure the width of our website, it will come to 1002px. To center this div on the page, we set the margins to auto. This essentially means both the left and right margins of this div MUST be the same value. The only way that this is possible is if the div is perfectly centered in the browser window. Additionally, we align the text back to the left, create a new positioning context, and add our container background image.

Header

Let's begin styling our header. Add the following code:

```
#header {
    border-left: 1px solid #5e463c;
    border-right: 1px solid #5e463c;
    overflow: hidden;
}

```

```
/* Navigation */

#container ul#nav {
    overflow: hidden;
    float: right;
    height: 111px;
    margin: 0;
    margin-top: -13px;
}

#nav li {
    float: left;
    font-size: 11px;
    font-family: arial;
    text-align: center;
    width: 112px;
    height: 111px;
    margin-right: 20px;
    list-style-type: none;
}

#nav li a {
    color: #c7b18d;
    height: 111px;
    display: block;
    padding-top: 86px;
    text-transform: uppercase;
    text-align: center;
    padding-left: 10px;
}

/* The icons for the navigation */

#nav li#li_home  {
    background: url(../images/icons/home.png) ;
```

```
}

#nav li#li_work {
    background: url(../images/icons/work.png);
}

#nav li#li_about {
    background: url(../images/icons/about.png);
}

#nav li#li_contact {
    background: url(../images/icons/contact.png);
}
```

Though it might initially seem confusing, mostly, we're grabbing the values from our PSD and pasting them into our CSS file. For instance, in the PSD, the header has a light brown border on the left and right.

Using the eyedropper tool, we grab that hex value, and then use the “border” property to set it. There’s nothing more to it than that.

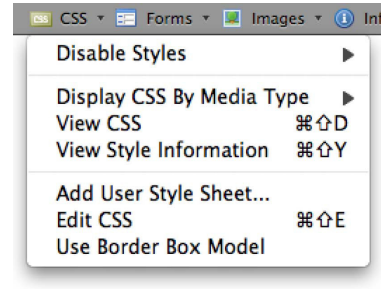
To show each of our navigation items as a line, we set the “float” property to “left”. Now, they’ll sit side by side. Next, we set the font attributes, add a bit of right-margin to provide some breathing room, and set the list-style to none.

Normally, we wouldn’t hard-code a width and height value here. However, because we’re using those navigation icons, we need to return to the PSD and measure the width and height of those images. I found them to be 112px wide and 111px high.

Remember when we applied a unique id to each list item?

```
<ul id="nav">
    <li id="li_home" class="selected"><a href="index.
html">Home</a></li>
    <li id="li_work"><a href="work.html">Work</a></li>
    <li id="li_about"><a href="about.html">About</a></li>
    <li id="li_contact"><a href="contact.html">Contact</
a></li>
</ul>
```

Now that will pay off. To target each specific navigation item, we simply use its id, and apply the appropriate background icon.

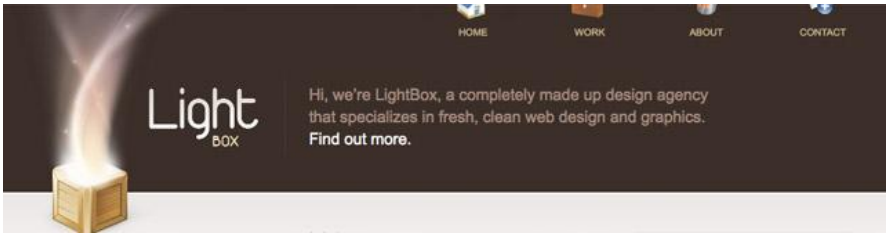


NOTE: As you can imagine, it would be awfully boring for you if I were to explain every single property. Instead, download the “Web Develper Extension” for Firefox, created by Chris Pederick.

"The Web Developer extension adds a menu and a toolbar to the browser with various web developer tools." – http://chrispederick.com/work/web-developer/.

With that extension installed, begin to experiment by removing a property to see how the site changes. For instance, what would happen if we remove the `#nav li` width and height properties completely? This is how I learned, and I recommend you do the same.

Styling The Main Banner



Let's continue on to our main banner, or slogan.

```
<div id="slogan">
    <h1><a href="index.html">Light Box</a></h1>
    <img src="images/box.png" alt="Light Box" />
    <p>Hi, we're LightBox, a completely made up design
agency that specializes in fresh, clean web design and
graphics. <a href="about.html">Find out more.</a></p>
</div><!-- end slogan-->
```

Append the following code to your stylesheet:

```
#slogan {
    height: 168px; /* Not typically a good idea to set
heights. In this case, it's necessary. */
    clear: both;
}

/* The Logo */

#slogan h1 {
    background: url(../images/logo.png) no-repeat;
    width: 124px;
    height: 67px;
     margin: 0 0 0 165px;
    position: relative;
    text-indent: -9999px;
    top: 50px;
}

#slogan h1 a {
    display: block;
    width: 124px;
    height: 67px;
}

/* The big light box */

#slogan img {
    position: absolute;
    top: -87px;
    left: -40px;
}

#slogan p {
    margin: -27px 200px 0 319px;
```

```
        padding-left: 27px;
        padding-top: 5px;
        padding-bottom: 5px;
        font-size: 18px;
        color: #a6a8a7d;
        border-left: 1px solid #4f3b33;
    }

    #slogan a {
        color: white;
    }
```

We begin by specifying a height for our slogan div. This is generally not desired, however, in this case, it makes our job much easier. Next, we use the "clear" property to ensure that our div clears (or begins below) any floated elements above. To gain a better understanding, using the Web Developer Toolbar, remove this property and review the difference.

The "Light Box" logo is contained within an H1 tag. Rather than hard coding the image into our HTML, we can use a background trick.

Fahrner's Image Replacement Technique

This technique is named after Todd Fahrner (a user interface technology consultant). To achieve our desired effect, we must perform the following:

1. Measure the width and height of the image, using the ruler tool in Photoshop – in this case,124x67.

2. Set the background property to our image's URL.

3. Set the width and height values equal to what we just measured.

4. Set “text-indent” equal to an enormous negative number, or -9999px. This will shift any text within the H1 tag off even the largest of screens, leaving only our image.

You’ll most often find this technique implemented to use specific fonts that aren’t available across all computers. By saving a heading as an image, we can then set it as a background and shift our text off the screen.

Continuing on to our image of the Light Box, we use absolute positioning to place it in the exact coordinates.

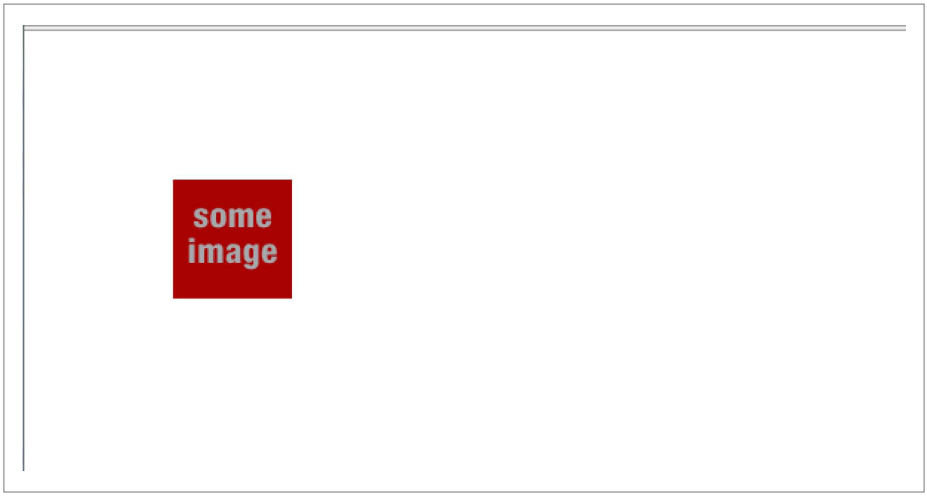
By taking it out of the flow of the document, we can then use the left, top, right, and bottom properties to position our element without affecting the layout.

What’s The Difference Between Absolute And Relative Positioning?

Maybe more than anything else, positioning can prove to be an unnecessarily confusing topic for beginning CSS designers. The best way to learn is to first tackle absolute positioning. Let’s assume that we have a blank html and CSS document. If we were to then place an image absolutely on the page – say 100px from

the top, and 100px from the left of the window's edges – we could write the following style:

```
img
{
position: absolute;
top: 100px;
left: 100p;
}
```



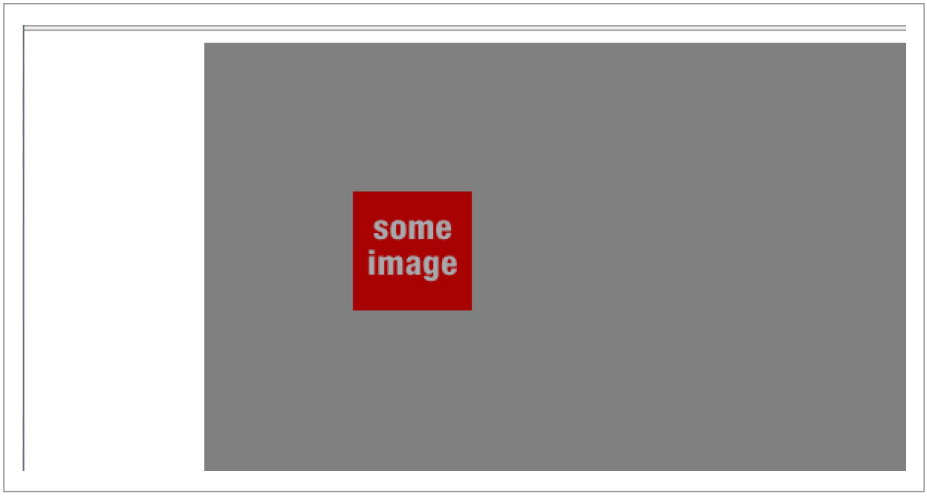
Absolutely positioned elements are positioned in relation to their closest positioned parent elements. In this case, where there are no relatively positioned elements on the page, the image will be positioned in relation to the window.

Now, imagine that we wrapped a relatively positioned div around our image.

```
<body>
  <div id="wrapper">
    <img src="#" alt="Some Image" />
  </div>
</body>
```

In order to set a positioning context, we must add “position: relative” to the styling of our parent div.

```
div#wrapper
{
position: relative;
background: gray; /*Just to see the borders.*/
height: 600px;    /*Because the image is absolutely
positioned, we need to force the height.*/
width: 770px;
margin: auto;
}
```



Now, when we absolutely position the image, it will be positioned "relative" to the "wrapper" division. Keep in mind that if we removed this property, the image would once again be placed in relation to the browser's window. The "position" property is key here.

Back to our project – we now need to position our slogan.

```
#slogan p {
    margin: -27px 200px 0 319px;
    padding: 5px 0 5px 27px;
    font-size: 18px;
    color: #a68a7d;
    border-left: 1px solid #4f3b33;
}
```

The font-size, color, and border-color are all taken from our PSD, as discussed previously. This is the easy part. To set our margins accurately, we must use the ruler tool in Photoshop to measure the distance between the edge of our site and the beginning of our text – on each side. Doing so – for me – returned -27px, 200px, 0, and 319px (top, right, bottom, left).

Main Content

It's now time to begin styling our main content, or the information within the `#main div`.

```
/***** MAIN CONTENT SECTION ******/

#main {
    padding: 0 36px 100px 347px;
    overflow: hidden;
    border-left: 1px solid white;
    border-right: 1px solid white;
```

```
}

#main #primaryContent {
    float: left;
    width: 334px;
}

#main div#secondaryContent {
    float: right;
    padding-top: 45px;
    width: 249px;
    position: relative;
}
```

Notice how we have quite a bit of white space on the left side of our site? To simulate this, we're going to use the padding property. Once again, return to your PSD and measure, with the ruler tool, the spacing between the edges of the website on each side. By my measurements, I found that there was 347px on the left side. By adding this value to our padding-left property, we'll create a nice open space that compliments our website well.

For our home page, we're using a simple two-column layout. To achieve this effect, we float our "primaryContent" div to the left, and the "secondaryContent" to the right. By specifying specific widths, taken from our PSD, our columns will then line up next to each other appropriately.

Working Example: Columns

In this quick example, we'll build an extremely simple mock layout, consisting of a header, main section, and footer. Within the main section, we'll nest two additional divs, which will serve as our two columns.

Within your code editor, create a basic HTML shell and then paste in the following code:

```
<div id="container">
<div id="header">

</div>
<div id="main">
<!-- main content-->
</div>

<div id="footer">
    <p>My footer info.</p>
</div>
</div><!--end container-->
```

As you can see from the code above, each section has received its own div. Additionally, we've wrapped our entire website in a "containing" div. This way, if we need to pick up our site, or box, and move it around, we only need to manipulate one div.

Creating the Columns

For the sake of simplicity, imagine that we've inserted the necessary text and images for our website.

The next step is to create our two floated columns. Within the "main" div, insert the following code:

```
<div id="main">

<div id="primary">
    <p> insert generic text here. </p>
</div>
<div id="secondary">
```

```
        <p>insert generic text here. </p>
    </div>

    </div>
```

CSS

Either within an external CSS file or the head tags of your document, add the following styles:

```
    #container {
        width: 960px;
        margin: auto;
        background: red; // pay attention to this one.
    }

    #header {
        min-height: 200px;
        background: yellow;
    }

    #main {
        background: green;
    }

    #primary {
        float: left;
        background: orange;
        width: 70%;
        margin-right: 5%;
    }

    #secondary {
        float: right;
        background: brown;
```

```
        width: 20%;
        margin-left: 5%;
    }

    #footer {
        background: black;
    }
```

By applying background colors to each of our divs, we'll gain an understanding of exactly how they line up. Some prefer to use borders; however, this can potentially change your layout due to the extra pixels. Play it safe and use background colors instead.

Notice how the "primary" and "secondary" divs were floated to the left and right, respectively? This one property ("float") is exactly what is used tocreate our columns.

How Does it Work?

Divs, in their natural state, are greedy. They take up as much space as possible, unless specifically told what to do (through the use of the "width" property).

By using the "float" property, we can force our divs to take up only as much width as we wish them to – in this case, 70% and 20%, respectively.

Three Columns

Now, we'll take things a step further and throw an additional column into the mix. Return to your structure and revise the "primary" div, like so:

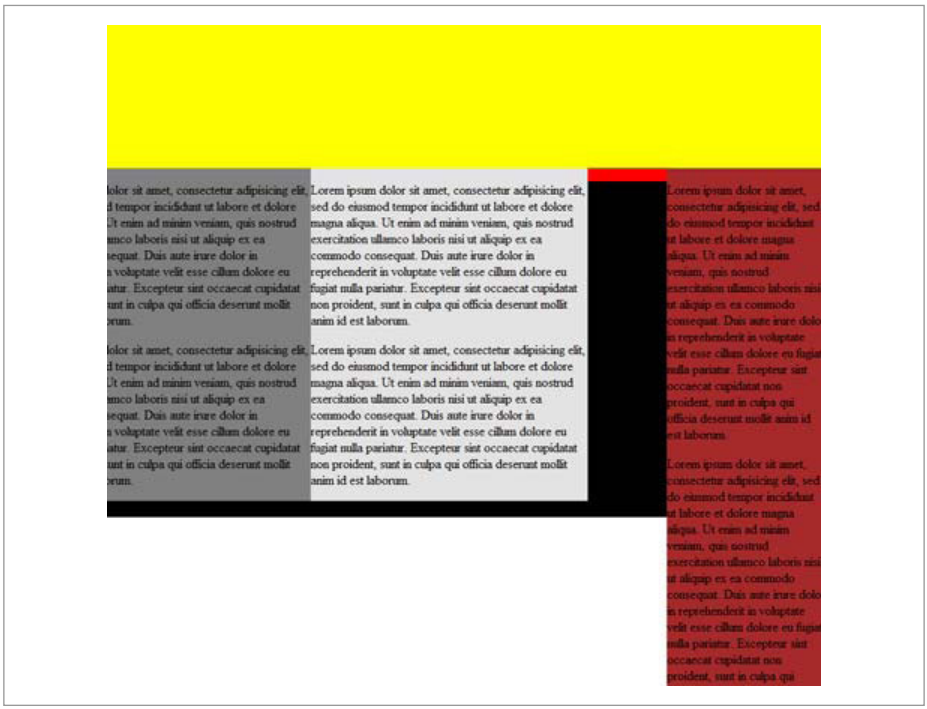
```
<div id="primary">
    <div id="col1">
            insert generic text here
    </div>
    <div id="col2">
            insert generic text here.
    </div>
</div><!--end primary-->
```

Rather than placing our content directly into the "primary" div, we'll insert our text into its children divs. Now return to your stylesheet and add the following:

```
#primary {
    float: left;
    background: orange;
    width: 70%;
    margin-right: 5%;
}
```

```
#col1 {
    float: left;
    background: gray;
    width: 50%;
}

#col2 {
    float: right;
    background: #e3e3e3;
    width: 50%;
}
```

Refresh your browser, and you'll see three side-by-side columns.



If you're like me, your initial instinct, in this scenario, would have been to simply create a third div, or `#primary`, `#secondary`, `#thirdly divs`. While this method does work, you'll find that it can be a bit difficult to line them up perfectly. This is because floats are very tricky things. **The addition of even a single pixel can potentially destroy a layout.** Considering this, the method demonstrated above will be your best option in most situations.

Here's The Thing about Floats

Now you surely must have noticed something strange occur once you began floating the divs. What happened to the container div's red background when we floated its children? Why did the red abruptly end right below the header? Since the container is a wrapper, shouldn't we see a red background instead of white?

Secondly, why is the footer (the black) taking up so much space, and why is it in the wrong place? The black should be BELOW the main content? What's the problem?

When you float an element, you're taking it out of the flow of the document. In our situation, we've floated BOTH elements within our "main" div. In other words, they're essentially taking up no space. As a result, the red background prematurely ends, and the footer jumps to the top.

Fixes

Over the years, there have been many fixes for this problem, most of which are still in use today. Let's review the three most common:

Fix 1: Clear:both

Return to your HTML structure. Just before the closing “main” div, append the following empty div:

```
        <div style="clear: both;" />
    </div><!--end main-->
```

By applying the “clear” property, we can force the “main” div to clear (or contain) its floated children. This method definitely works, but there’s just one problem: we’ve added a bit of unsemantic markup. Is there a way to accomplish this without adding useless divs? Yep!

Fix 2: Clearfix Hack

This second fix uses a bit of advanced CSS to clear any floats. Consider the following styles:

```
    #main:after {
        content: ".";
        display: block;
        height: 0;
        clear: both;
        visibility: hidden;
    }
```

Here, we’re using the “after” pseudo-class to append content after the element from which it was referenced, in this case, the “main” div. Most importantly, take note of the “content” property. It might look a bit complicated at first, but it’s actually quite simple. We’re simply adding a period. Now of course, we don’t want this period to display on the page. To compensate, we’ve set the “visibility” property to “hidden.” Now that we’ve added a bit of content after our “main” div, we can simulate the actions from Fix 1: set “clear”

to “both”, and “display” to “block”. However, note that pseudo classes, such as “:after” will not be understood by Internet Explorer 6 and below.

That’s all there is to it. This will work, but it’s still somewhat tedious. Luckily, we can work around this entire issue simply by adding one style! It’s a wonder how this final method was only popularized a couple of years ago.

Fix 3: Overflow: Hidden

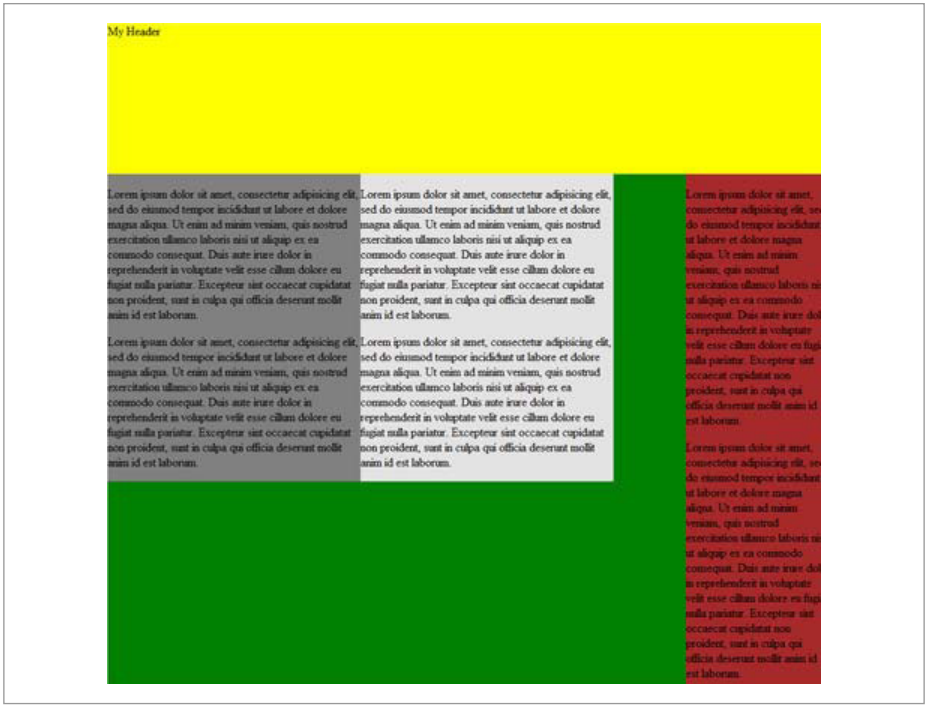
Remove all of your code from Fixes 1 and 2 – we’ll no longer need it. In their place, add one property to the “main” div.

```
#main {
    ..other styles
    overflow: hidden;
}
```

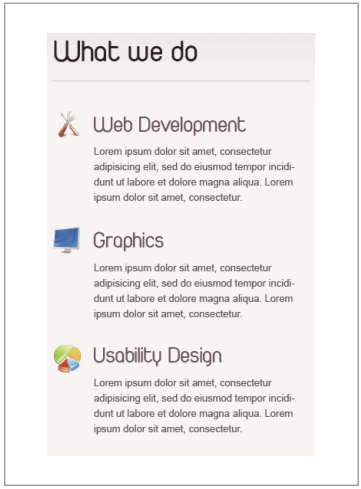
Believe it or not, this one property solves our problem. If you’ll remember back to the first few pages of this book, I referenced the fact that, sometimes, a solution makes little sense, if any. This is absoluely true in this case. Nevertheless, it works.

ROCK* TIP

If you float more than one element within a parent, add “overflow: hidden;” to the parent’s styling. This will force the wrapping element to contain the floated elements.



The Three Headings



If you've come this far, you're doing very well! This stuff is tricky! We must now style our three headings. When we converted our PSD, we saved these as one large image. That will pay off now when we use CSS sprites to display only our desired portion of the image.


```
#primaryContent ul h3 {
    text-indent: -9999px;
    height: 34px;
    margin-bottom: 13px;
}

/* Sprites are used here. The same image contains all of
the headings. We use the background-position property to
shift the image upward. */

/* What We Do */
#primaryContent li#first h3 {
    background: url(../images/whatWeDoHeadings.png) no-
repeat 0 0;
}

/* Graphics */
#primaryContent li#second h3 {
    background: url(../images/whatWeDoHeadings.png) no-
repeat 0 -34px;
}

/* Usability Design */
#primaryContent li#third h3 {
    background: url(../images/whatWeDoHeadings.png) no-
repeat 0 -68px;
```

```
}

#first p, #second p, #third p {
    padding-left: 50px;
}
```

We'll be implementing the same background technique that we learned earlier. Most importantly, take note of the fact that we set the height of our heading three tag to "34px", or one-third of the entire height of our image. This will, in effect, cut off the other headings. Additionally, we've set a margin-bottom of 13px to add some breathing room.

Next, we need to target each heading individually. Because we created a unique id for these headings when we created our markup, we have the ability to do so. Each heading will use the exact same image as its background, but the browser will only need to load it once. This is how we save bandwidth. The key here is that we're using background positioning to shift the image. Notice how the graphic has been shifted upward -34px – or the height of our h3 tags exactly. The "Usability Design" heading will have that number doubled to 68px accordingly as well. It's as simple as that. If your eyes are glossing over as you read this, I don't blame you! Use the web developer extension, and just play around with the positioning to better understanding.

Finally, we need to add 50px worth of padding-left to our paragraphs in order to shift them enough so that they're directly underthe heading, rather than under the icon.

Blockquote

Let's now style our testimonials section. This shouldn't be too difficult. Here's the HTML code:

```
<div id="secondaryContent">
    <blockquote>
            <p>Lightbox did an amazing job with our website.
From start to finish they were professional, fast and
totally fictitious. </p>
            <p>It was amazing how a company that doesn't
even exist could put together such an amazing site for us.
Thank you Lightbox!</p>
    </blockquote>
    <cite title="MadeUpPeople.com">John Rushmore
<span>MADEUPPEOPLE.COM</span></cite>
```

```
        <div class="arrow"></div>
    </div><!-- end secondaryContent-->
```

Now paste in the following CSS into your stylesheet.

```
    blockquote {
        background: #e6e2df;
        color: #7f6357;
        padding: 23px;
        font: italic 20px/27px georgia;
    }

    blockquote p {
        font-size: 20px;
        margin-bottom: 1em;
        color: #7f6357;
        font-family: georgia;
    }

    #secondaryContent  cite {
        font: bold normal 15px arial;
        color: #2f231e;
        float: right;
        padding-right: 20px;
        padding-top: 35px;
        font-weight: bold;
    }

    #secondaryContent cite span {

        font-size: 10px;
        line-height: 10px;
        display: block;
        color: #89451f;
    }
```

Hopefully, you're beginning to learn that converting a design from a PSD to a working website is partially a process of copy and paste. Take the blockquote, for example; the background-color, text color, font-size, font-family, and padding are 100% taken from the PSD. It's just a simple matter of using the eyedropper, ruler, and text tool.

We must style the author's website URL separately. This is why the address is wrapped within a span tag. We can easily target that specific text and style it accordingly by using the following selector: `#secondaryContent` cite span – or, find the span tag that is within a cite, which is within an element with an id of "secondaryContent". Once again, these values were taken directly from their counterparts in the PSD.

CSS Shapes

Did you know that the illusion of a shape can be created with 100% CSS? We're going to use this technque to create the arrow at the bottom of our blockquote.

The key is with the extra div that we added to our markup.

```
<div class="arrow"></div>
```

Return to your stylesheet, and just below the blockquote selectors, add the following:

```
/* The CSS arrow! */

.arrow {
 width: 0;
 height: 0;
 line-height: 0;
 border-top: 30px solid #e3e3e3;
 border-left: 60px solid #f7f5f4;
 border-right: 10px solid #f7f5f4;
 position: absolute;
 right: 30px;
}
```

Reload index.html in your browser, and you'll see your arrow! Actually, we're still working with a square; nothing has change. However, by setting the width and height properties equal to zero, we can then create our triangle using the border properties. If we set the color of our borders equal to the background color of our page, those sections will appear to not exist, thus creating the illusion of a shape.

To better unerstand, look what happens when we change the border colors.

It's still a rectangle, but when we change the blue and red equal to the background of the body (the off-white), they seemingly disappear. To learn more about this neat trick, I've created an in depth video tutorial on the subject. If you are interested, be sure to watch it. (http://net.tutsplus.com/videos/screencasts/fun-with-css-shapes/)

Note: this will not work in Internet Explorer 6. As we've learned, a website doesn't need to look identical in every single browser. For these users, we'll simply hide the arrow entirely – no big deal.

Recent Portfolio

It's time to begin styling our recent portfolio items. Here's the HTML code:

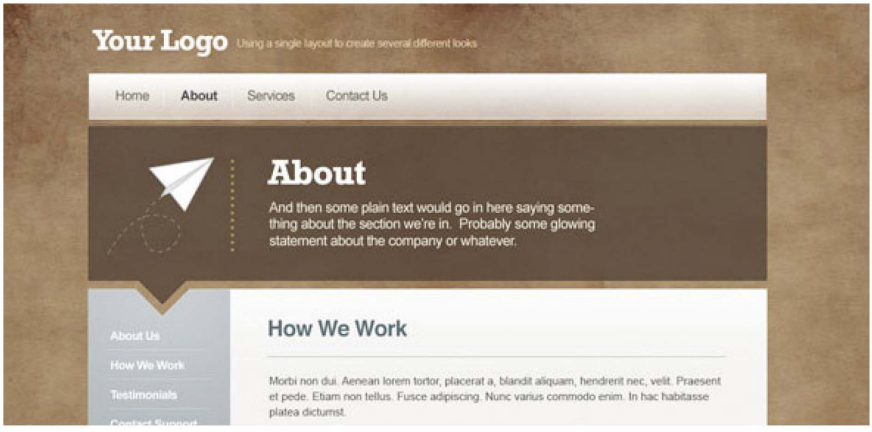
```
<div id="recentWork">
    <h2>Recent Work</h2>
    <ol>
            <li class="chosen"><a href="images/recentWork/
recentWork1.jpg" title="A design layout I created for
Psdtuts+">1</a></li>
            <li><a href="images/recentWork/recentWork2.jpg"
title="A pastel layout I created for Psdtuts+">2</a></li>
            <li><a href="images/recentWork/recentWork3.jpg"
title="A grass layout I created for Psdtuts+">3</a></li>
            <li><a href="images/recentWork/recentWork4.jpg"
title="A dark layout I created for Psdtuts+">4</a></li>
    </ol>

    <div class="image_display"><!-- needed to apply a
border to both the image and the edges around the
padding ->
            <img src="images/recentWork/recentWork1.jpg"
alt="Recent Work" />
```

```
        </div>
        <p><strong>Web Design Week Layout:</strong> A design
    layout I created for Psdtuts+</p>
    </div><!-- end recent work-->
```

Before

Recent Work

1. 1
2. 2
3. 3
4. 4



Web Design Week Layout: A design layout I created for Psdtuts+

Insert the following CSS:

```
#main div#recentWork {
    clear: both;
    position: relative;
    border-top: 1px solid #cec5c0;
}
```

```
#subpage #container #recentWork {
    border-top: none;
}

#subpage #recentWork h3 {
    font-size: 24px;
    color: #2f231e;
}

#recentWork h2 {
    border: none;
    margin-bottom: 0;
    padding: 28px;
    margin-top: 11px;
}

#recentWork ol {
    position: absolute;
    right: 0;
    top: 45px;
}

#subpage #recentWork ol {
    top: 0;
}

#recentWork ol li {
    width: 16px;
    height: 16px;
    display: inline;
}

#recentWork ol li a {
    color: white;
    background: #bc535a;
```

```
        width: 100%;
        height: 100%;
        padding: 5px 10px;
        -moz-border-radius: 100px;
        -webkit-border-radius: 13px;
    }

    #recentWork ol li.chosen a, #recentWork ol li a:hover{
        background: #a37a7e;
    }

    #recentWork ol li a:hover {
        text-decoration: none;
    }


    #recentWork p {
        text-align: right;
        font-size: 13px;
        font-style: italic;
        color: #4e4d4d;
        margin-top: 10px;
    }

    #subpage #recentWork p {
        text-align: left;
        line-height: 18px;
    }

    div.image_display {
        padding: 11px;
        border: 1px solid #d6cec9;
        background: white;
    }
```

```
.image_display img {
    border: 1px solid #dbcec7;
}
```

We begin by styling our wrapping `#recentWork div`. Because the blockquote above was floated to the right, we must be sure to clear the elements above first. Next, it's generally a smart idea to create a new positioning context for your wrapping divs. Finally, we set a 1px border-top – the color being taken from the PSD, of course.

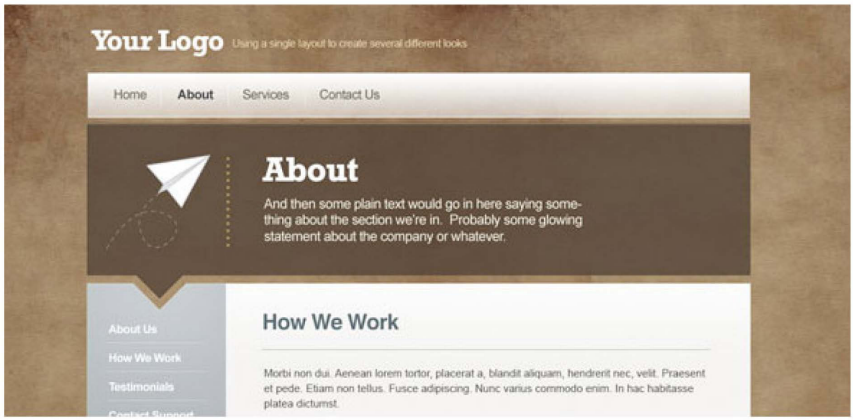
Continuing on to our heading two tag "Recent Work", we once again implement Fahrner's background technique (Page 64) and apply our image as a background.

The ordered list will house our numbered items. When the user clicks on a number, we'll ultimately use jQuery to change the image. To position them exactly where we want, we once again use "absolute" positioning. Thankfully, we created a new positioning context earlier – otherwise, our top, right, bottom, and left values would be placed in reference to the `#main div` instead. A quick use of the "ruler tool" shows that we need 45px of spacing from the top border.

After

Recent Work

1. 1
2. 2
3. 3
4. 4



Web Design Week Layout: A design layout I created for Psdtuts+

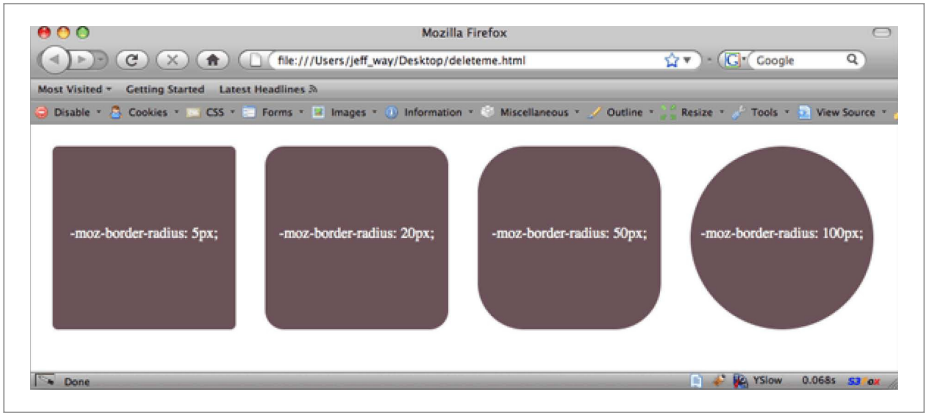
Rounded Corners

Though we could have used a background image to create the brown circular backgrounds, we'll instead use a neat browser styling trick to achieve this effect. Note that this will only work in Firefox, Safari, and Chrome – not Internet Explorer. In that browser, the circles will become squares, which is an acceptable change.

ROCK* TIP

It's okay if your website doesn't look 100% the same in every browser.

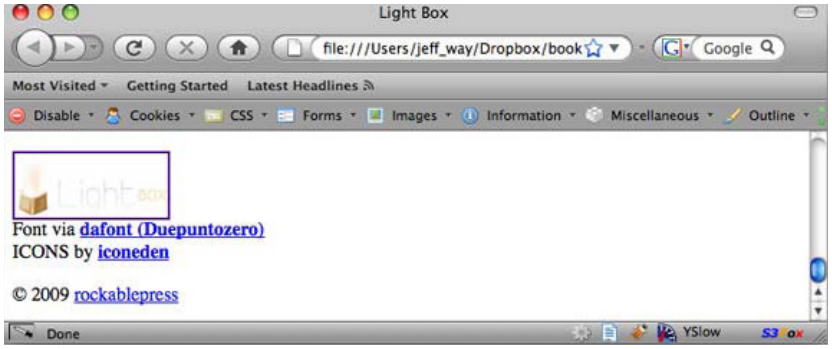
!



To create our rounded edges, we use the **-moz-border-radius**, and **-webkit-border-radius** properties to target Firefox and Safari / Chrome, respectively. These two values are determined slightly differently. Typically, these properties are used to apply a subtle curvature to elements. In our case, we wish to create a complete circle. 100px for Firefox, and 13px for Safari should do the trick. Nifty, ay? Let's also apply a bit of padding to give our numbers some breathing room.

Footer

The final step for our home page is to style the footer.



HTML

```
</div><!--end container-->
<div id="footerWrap"><!-- needed to extend the brown
background for the entire screen -->
    <div id="footer">
            <div id="footerLogo">
                    <a href=""><img src="images/footerLogo.
png" alt="Light Box" /></a>
            </div>
                    Font via <strong><a href="http://www.
dafont.com/duepuntozero.font">dafont (Duepuntozero)</a>
</strong><br />
                    ICONS by <strong><a href="http://
iconeden.com/icon/bright-free-stock-iconset.
html">iconeden</a></strong>
                    <p id="footerInfo">&copy; 2009  <a
href="http://rockablepress.com/">rockablepress</a></p>
            </div><!-- end footer -->
</div><!-- end footerWrap-->
```

CSS

```
/* Footer */

/* The footer div must be wrapped with this #footerWrap
div in order to extend the brown background for the entire
width of the screen. */
#footerWrap {
    border-top: 1px solid white;
    background: #32251f;
}

#footer {
    overflow: hidden;
```

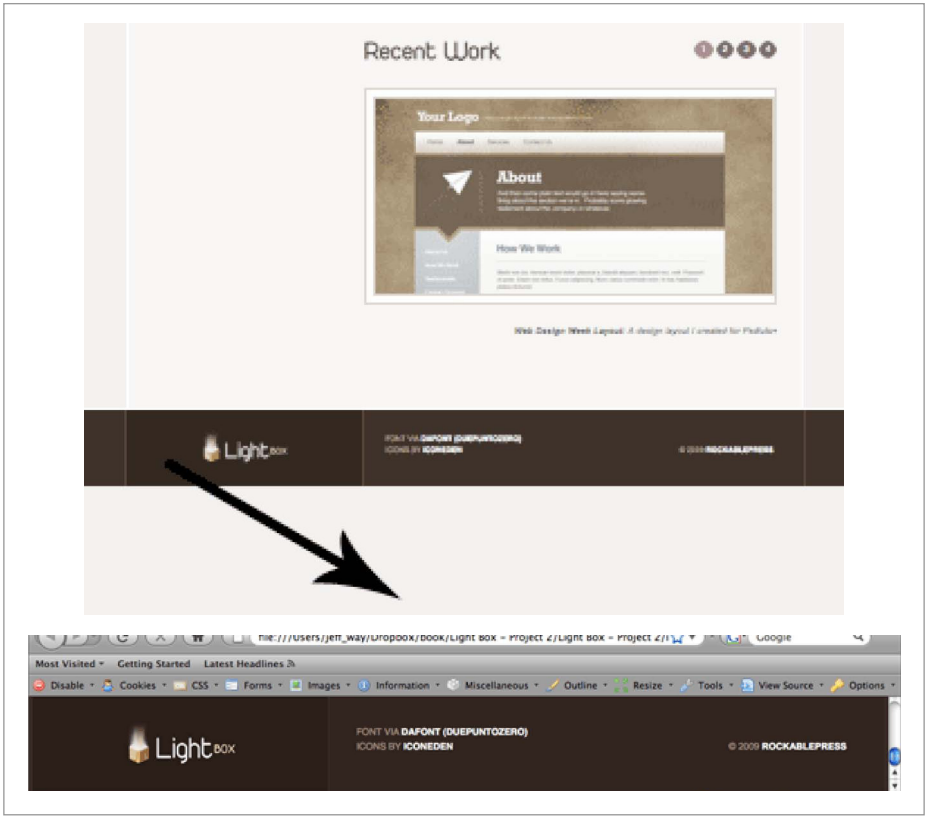
```
        color: #a6a7d;
        width: 1002px;
        margin: auto;
        text-align: left;
        padding-top: 35px;
        font-size: 11px;
        line-height: 17px;
        text-transform: uppercase;
        border-right: 1px solid #48362d;
        border-left: 1px solid #48362d;
    }

    #footer p {
        color: #a6a7d;
        font-size: 11px;
    }

    #footer #footerLogo {
        float: left;
        background: #291e18;
        width: 348px;
        text-align: center;
        padding-top: 2.5em;
        padding-bottom: 3em;
        margin-right: 32px;
        margin-top: -35px;
        border-right: 1px solid #48362d;
    }

    #footer strong a, #footer p a {
        color: white;
        font-weight: bold;
    }

    #footer p#footerInfo {
        float: right;
```

```
        padding-right: 45px;
        margin-top: -17px;
    }
```

To obtain a repeating brown background at the bottom of our site, we can't place our footer with the "`container`" div. If we did so, it would be cut off beyond 1004px (the width of our website). However, we still want our footer to appear as if it's connected to our website. The solution is to apply a wrap to our footer, called "footerWrap" in the code. We'll set a background color and leave off the width. This is what allows the background to repeat for the entire width of the window. Then, for the "`#footer`" child div, we'll apply the exact same values as we did for our `#container` element.

```
    #footer {
        overflow: hidden;
        color: #a68a7d;
        width: 1002px;
        margin: auto;
        text-align: left;
        padding-top: 35px;
        font-size: 11px;
        line-height: 17px;
        text-transform: uppercase;
        border-right: 1px solid #48362d;
        border-left: 1px solid #48362d; }
```

The remaining properties are once again taken directly from our PSD. You should be a pro at this by now.



Sticky Footers

There's one slight problem. If you're viewing the site in Firefox, press Control or Command – (Minus) to zoom out a few clicks. Notice how the footer jumps up?

It's not a huge deal, but we should definitely fix it. **Footers that are "locked" to the bottom of the browser window are known as "sticky footers."**

To achieve this effect, we'll need to add just a few additional styles within our CSS file.

Just before the beginning “body” selector, append the following code:

```
Html, body {
Height: 100%;
}
```

This instructs the browser to ensure that the html and body elements take up the entire height of the user’s screen.

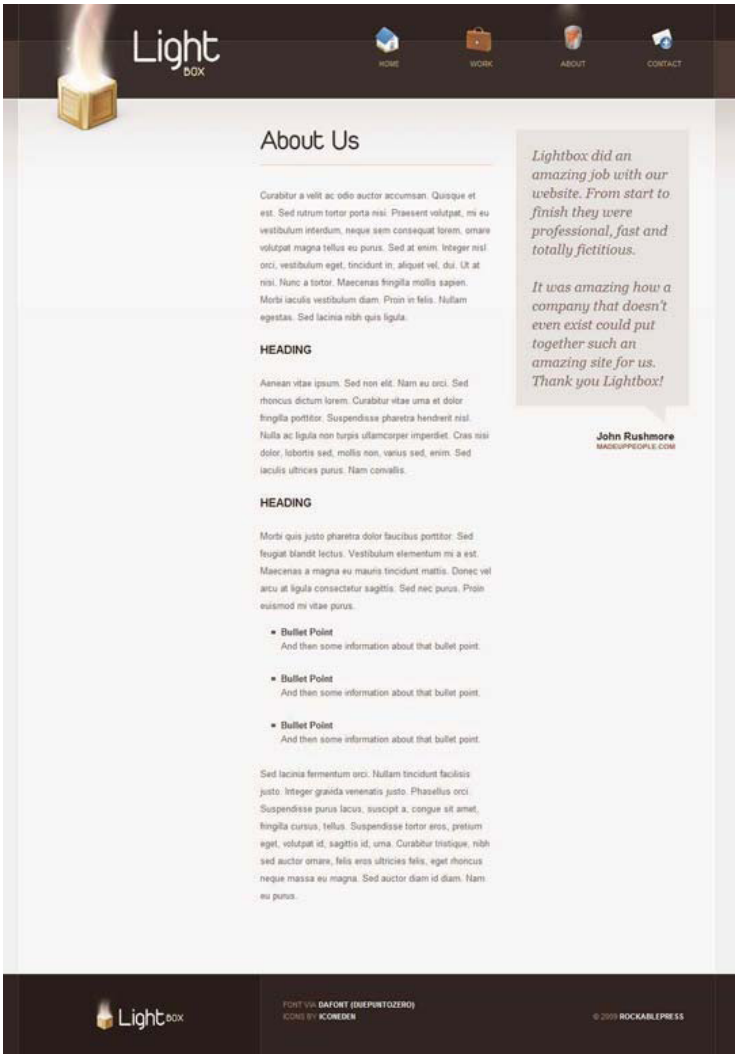
Find your “container” selector, and insert these new styles.

```
#container {
… other styles
min-height: 100%;
height: auto !important; /* for IE 6 */
height: 100%;  /* for IE 6 */
}
```

That’s all there is to it. Save the file, return to your browser, and reload the age. Voila!

About Page

Now that we've completed our Home page, it should prove much easier to style the remaining pages. Most of the coding has already been completed! It's just a matter of replacing text. We'll begin with the "About" page.



The Markup

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//
EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
<link rel="shortcut icon" href="images/favicon.ico" />
<link rel="stylesheet" href="css/reset.css" />
<link rel="stylesheet" href="css/default.css" />

<body id="subpage">

<div id="container">
    <div id="header">
            <ul id="nav">
                    <li id="li_home"><a href="index.
html">Home</a></li>
                    <li id="li_work"><a href="work.
html">Work</a></li>
                    <li id="li_about" class="selected"><a
href="about.html">About</a></li>
                    <li id="li_contact"><a href="contact.
html">Contact</a></li>
            </ul>

            <div id="slogan">
                    <h1><a href="index.html">Light Box</a>
</h1>
                    <img src="images/box.png" alt="Light Box"
/>
            </div><!-- end slogan-->
    </div><!-- end header-->
```

```
      <div id="main">

            <div id="primaryContent">
                  <h2 id="aboutUs">About Us</h2>

                  <p>
                        … filler text
                  </p>

                  <h4>Heading</h4>

                  <p>
                        … filler text
                  </p>
                   <h4>Heading</h4>

                  <p>
                        … filler text
                  </p>

                  <ul>
                        <li><span>Bullet Point</span>And
  then some information about that bullet point.</li>
                        <li><span>Bullet Point</span>And
  then some information about that bullet point.</li>
                        <li><span>Bullet Point</span>And
  then some information about that bullet point.</li>
                  </ul>
                  <p>
                        … filler text
                  </p>

            </div> <!-- end primaryContent-->

            <div id="secondaryContent">
                  <blockquote>
```

```
                        Lightbox did an amazing job with
our website. From start to finish they were professional,
fast and totally fictitious.
                        <br /><br />It was amazing how a
company that doesn't even exist could put together such an
amazing site for us. Thank you Lightbox!
                    </blockquote>
                    <cite>John Rushmore <span>MADEUPPEOPLE
COM</span></cite>
                    <div class="arrow"></div>
            </div><!-- end secondaryContent-->

    </div><!-- end main-->
 </div><!--end container-->

<div id="footerWrap"><!-- needed to extend the brown
background for the entire screen -->
    <div id="footer">
            <div id="footerLogo">
                    <a href=""><img src="images/footerLogo.
png" alt="Light Box" /></a>
            </div>
                    Font via <strong><a href="http://www.
dafont.com/duepuntozero.font">dafont (Duepuntozero)</a>
</strong><br />
                    ICONS by <strong><a href="http://
iconeden.com/icon/bright-free-stock-iconset.
html">iconeden</a></strong>
                    <p id="footerInfo">&copy; 2009  <a
href="http://rockablepress.com/">rockablepress</a></p>
            </div><!-- end footer -->
</div><!-- end footerWrap-->

</body>
</html>
```

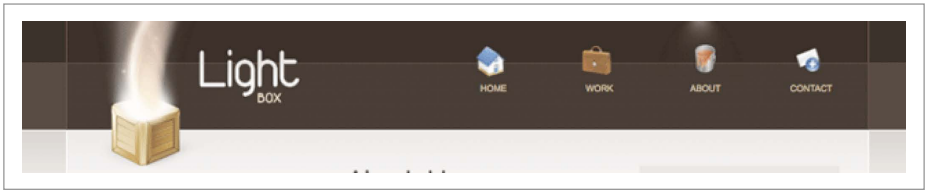
At first glance, this might seem a bit overwhelming. Don't worry, it's more or less the same code as our Home page. Other than the text, there are only two differences:

1. We've removed the "Recent Work" section that existed on the home page. It's not necessary here.

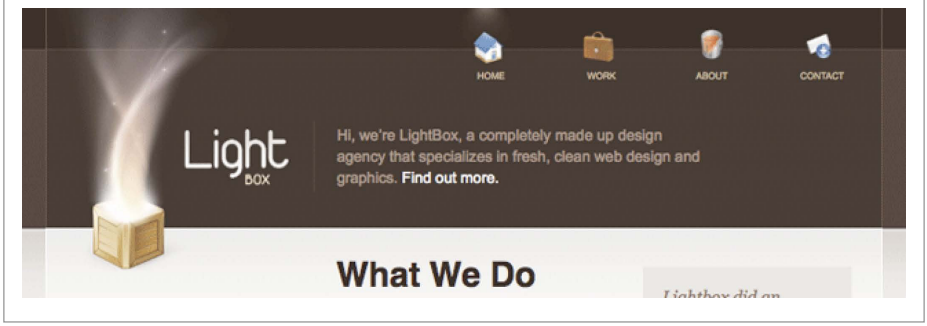
2. We've added an id to the body tag: "subpage."

Notice how the background is different on our subpage? It's much shorter.

About Page Header (subpage)



Home Page Header



The Body ID Trick

A quick glance at the "body" selector in our CSS file shows the following:

```
body {
    background: #f1efee url(../images/bg.png) repeat-x;
    text-align: center;
    font-family: helvetica, arial, sans-serif;
    line-height: 25px;
}
```

Most notably, we added a background image here. So how can we set one background image on the home page, and a different one on other pages? The most efficient method is to use the "body id" trick.

ROCK* TIP

By applying a unique id to our body tag, we can then style the element differently on a page-by-page basis.

Return to your default.css file, and just under the "body" styling, append the following rule:

```
body#subpage {
    background: #f1efee url(../images/bg_alt.jpg) repeat-x;
}
```

Save the file and then reload your about.html page. At the moment, the background should be completely white. That's because we need to slice out our new image!

Return to Photoshop and open “about.psd”. As you did before:

1. Flatten the PSD *(Layer > Flatten Image)*.

2. Select a long and narrow strip ofthe background.

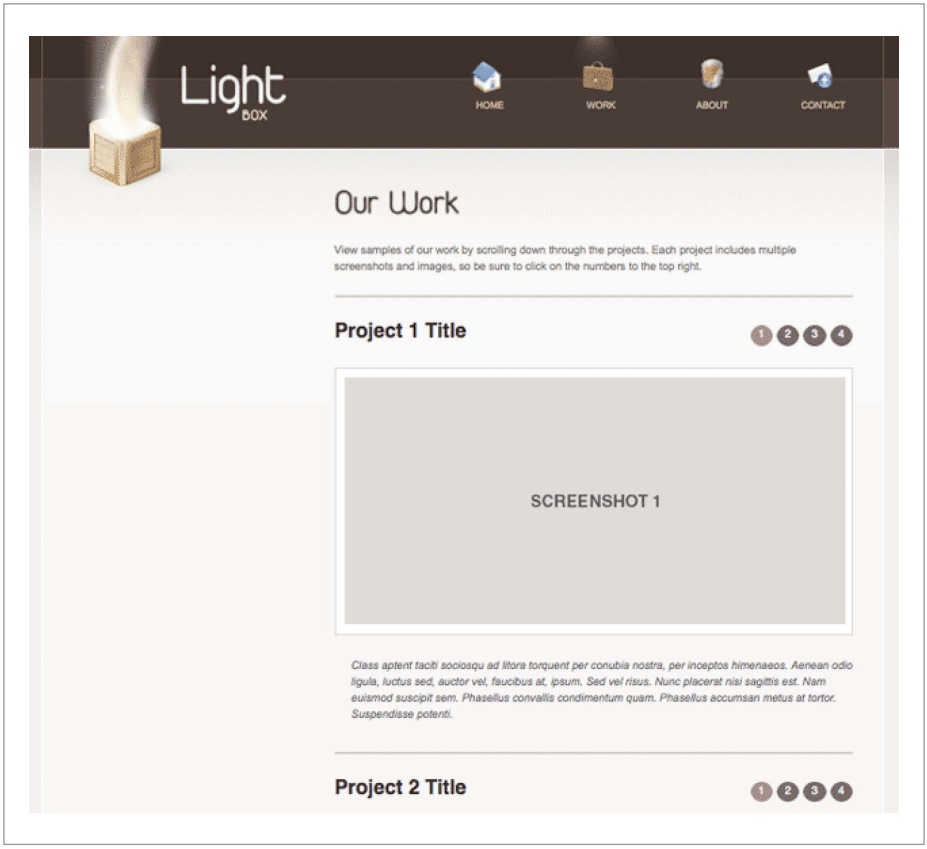
3. Repeat “The Pattern” (Page 36) and save it into your images folder as “bg_alt.jpg”.

4. Return to your browser and reload the page!

Because we’ve applied a unique body id to our “about page”, and have targeted that respective id in our CSS file, we can style it however we wish.

We've already taken care of most of our styling. Having done so, our "About" page is now finished.

Work Page



On to the "Work" page. Create a new file in your project called "work.html". Paste the following html:

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
    <title>Light Box</title>
    <link rel="stylesheet" href="css/reset.css" />
    <link rel="stylesheet" href="css/default.css" />
</head>
<body id="subpage" class="single">
  <div id="container">
    <div id="header">
            <ul id="nav">
                    <li id="li_home" class="selected">
<a href="index.html">Home</a></li>
                    <li id="li_work"><a href="work.
html">Work</a></li>
                    <li id="li_about"><a href="about.
html">About</a></li>
                    <li id="li_contact"><a href="contact.
html">Contact</a></li>
            </ul>
            <div id="slogan">

                    <h1><a href="index.html">Light Box</a>
</h1>
                    <img src="images/box-trans.png"
alt="Light Box" />

            </div> <!-- end slogan-->

    </div> <!-- end header-->

    <div id="main">

```

```
            <div id="primaryContent">

                    <h2> Our Work </h2>

                    <p>
                    View samples of our work by scrolling
down through the projects. Each project includes multiple
screenshots and images,
                    so be sure to click on the numbers to the
top right.
                    </p>

                    <div id="recentWork">
                            <h3>Project 1 Title</h3>

                            <ol>
                            <li class="chosen"><a
href="images/recentWork/recentWork1.jpg" title="Web Design
Week Layout: A design layout I created for Psdtuts+">1</
a></li>
                            <li><a href="images/recentWork/
recentWork2.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">2</a></li>
                            <li><a href="images/recentWork/
recentWork3.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">3</a></li>
                            <li><a href="images/recentWork/
recentWork4.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">4</a></li>
                    </ol>

                    <div class="image_display">
                            <img src="images/recentWork/
recentWork1.jpg" alt="Recent Work" />
                    </div> <!-- end image_display-->
```

```
                        <p>
                                Class aptent taciti
sociosqu ad litora torquent per conubia nostra, per
inceptos himenaeos. Aenean odio ligula,
                                luctus sed, auctor vel,
faucibus at, ipsum. Sed vel risus. Nunc placerat nisi
sagittis est. Nam euismod suscipit
                                sem. Phasellus convallis
condimentum quam. Phasellus accumsan metus at tortor.
Suspendisse potenti.
                        </p>

                        <hr />

                        <h3>Project 2 Title</h3>

                        <ol>
                        <li class="chosen"><a
href="images/recentWork/recentWork1.jpg" title="Web Design
Week Layout: A design layout I created for Psdtuts+">1</
a></li>
                        <li><a href="images/recentWork/
recentWork2.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">2</a></li>
                        <li><a href="images/recentWork/
recentWork3.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">3</a></li>
                        <li><a href="images/recentWork/
recentWork4.jpg" title="Web Design Week Layout: A design
layout I created for Psdtuts+">4</a></li>
                    </ol>

                    <div class="image_display">
                        <img src="images/recentWork/
recentWork1.jpg" alt="Recent Work" />
                    </div> <!-- end image_display-->
```

```
                        <p>
                                Class aptent taciti
sociosqu ad litora torquent per conubia nostra, per
inceptos himenaeos. Aenean odio ligula,
                                luctus sed, auctor vel,
faucibus at, ipsum. Sed vel risus. Nunc placerat nisi
sagittis est. Nam euismod suscipit
                                sem. Phasellus convallis
condimentum quam. Phasellus accumsan metus at tortor.
Suspendisse potenti.
                        </p>
                </div>


            </div> <!-- end primaryContent-->

    </div> <!--end main-->

  </div> <!-- end container-->

  <div id="footerWrap">

    <div id="footer">

            <div id="footerLogo">
                    <a href="index.html"><img src="images/
footerLogo-trans.png" alt="Light Box" /></a>
            </div>
            Font via <a href="http://www.dafont.com/
duepuntozero.font">dafont (Duepuntozero)</a><br />
            ICONS by <a href="http://www.iconeden.com/
">iconeden</a>

            <p id="footerInfo"> copyright (c) 2009  <a
href="http://www.rockablepress.com">rockablepress</a> </p>
    </div> <!-- end footer-->
```

```
    </div> <!-- end footerWrap-->

    <script src="http://ajax.googleapis.com/ajax/libs/
  jquery/1.4/jquery.min.js" type="text/javascript"
  charset="utf-8"></script>
    <script type="text/javascript" src="js/scripts.js">
  </script>

    <script type="text/javascript">
      Cufon.now();
    </script>

  </body>
  </html>
```

The "Work" page raises a new challenge. Up until now, our pages have had two columns within the main content. However, now we only have one column. As a result, we need to revisit our CSS file and make a few updates and changes.

We must find a way to make the "primaryContent" div a stated width for the Home and About pages, but 100% wide for the single column pages.

NOTE: Notice how we don't return to the HTML to make adjustments to the "presentation" of the page. That's a job for CSS!

Let's use the "body" trick again! Find the "body" tag and add a new class attribute.

```
  <body id="subpage" class="single">
```

Next, open your CSS file, find the `#primaryContent` declaration, and just under it, append a new style.

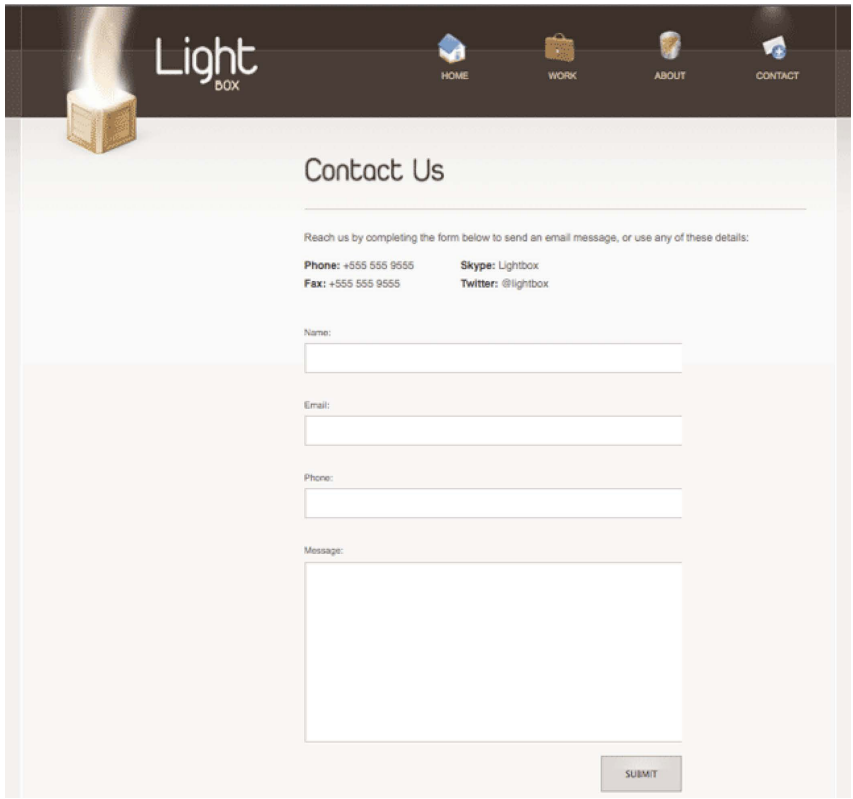
```
#main #primaryContent {
    float: left;
    width: 334px;
}
/* This rule is for the pages with only one column in the
main section. */
body.single #main #primaryContent {
    width: 100%;
}
```

By targeting the primaryContent element that is within a body tag with a class of “single,” we can adjust the width property only for this page. Easy.

As I’ve recommended before, it’s worth taking a moment to experiment with removing these styles directly in your browser, with the web developer extension for Firefox. That’ll give you a better understanding of what each declaration does. Mostly, we’re just matching the spacing with our PSD.

Contact Page

You've done so well if you've made it this far. We now must create our final page: "contact.html".



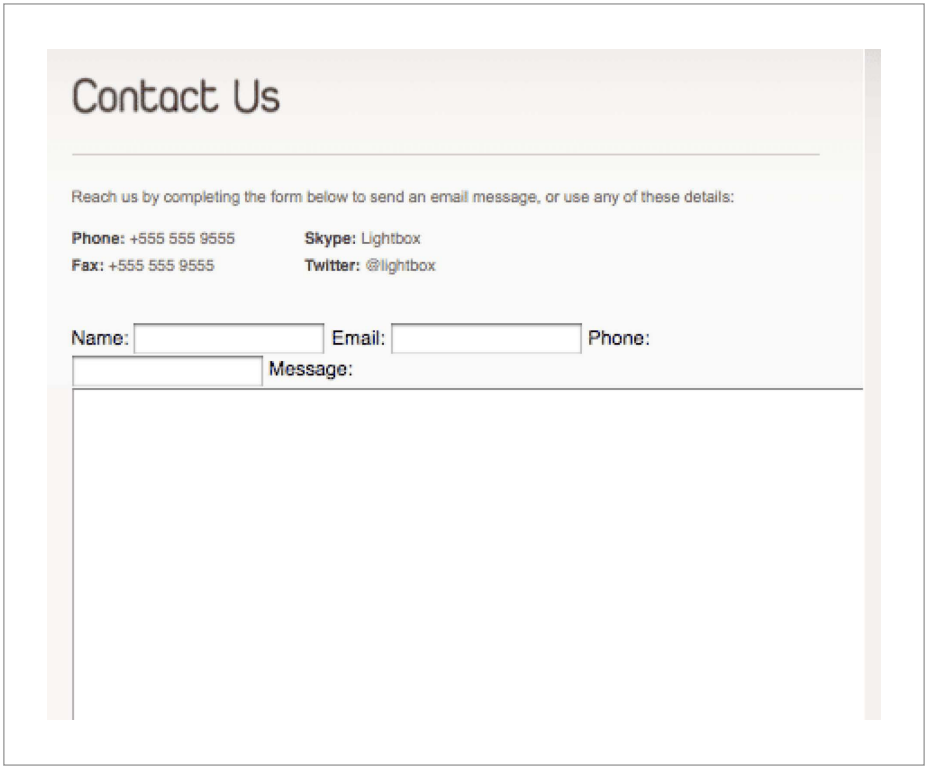
As always, paste in the following code:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//
EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.
dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=iso-8859-1" />
<link rel="shortcut icon" href="images/favicon.ico" />
<link rel="stylesheet" href="css/reset.css" />
<link rel="stylesheet" href="css/default.css" />

<title>Light Box</title>
</head>

<body id="subpage" class="single">

<div id="container">
    <div id="header">
            <ul id="nav">
                    <li id="li_home"><a href="index.
html">Home</a></li>
                    <li id="li_work"><a href="work.
html">Work</a></li>
                    <li id="li_about"><a href="about.
html">About</a></li>
                    <li id="li_contact" class="selected"><a
href="contact.html">Contact</a></li>
            </ul>

            <div id="slogan">
                    <h1><a href="index.html">Light Box</a></
h1>
                    <img src="images/box.png" alt="Light Box"
/>
            </div><!-- end slogan-->
    </div><!-- end header-->
```

```
    <div id="main">

        <div id="primaryContent">
            <h2 id="contactUs">Contact Us</h2>

            <p>
                Reach us by completing the
form below to send an email message, or use any of these
details:
            </p>

            <ul id="info">
                <li><strong>Phone: </strong> +555
555 9555</li>
                <li><strong>Skype: </strong>
Lightbox</li>
                <li><strong>Fax: </strong> +555
555 9555</li>
                <li><strong>Twitter: </strong> @
lightbox</li>
            </ul>
        <form action="contact.html" method="get"
accept-charset="utf-8">
                <label for="name">Name: </label>
                <input type="text" name="name"
value="" id="name" />

                <label for="email">Email: </
label>
                <input type="text" name="email"
value="" id="email" />
                <label for="phone">Phone: </
label>
                <input type="text" name="phone"
value="" id="phone" />
                <label for="message">Message: </
```

```
label>
                            <textarea name="message"
id="message" rows="216" cols="452"></textarea>

                            <input type="submit"
name="submit" value="Submit" />
                    </form>

            </div><!--end primaryContent-->

    </div><!-- end main-->

</div><!--end container-->

<div id="footerWrap"><!-- needed to extend the brown
background for the entire screen -->
    <div id="footer">
            <div id="footerLogo">
                    <a href=""><img src="images/footerLogo.
png" alt="Light Box" /></a>
            </div>
                    Font via <strong><a href="http://www.
dafont.com/duepuntozero.font">dafont (Duepuntozero)</a>
</strong><br />
                    ICONS by <strong><a href="http://
iconeden.com/icon/bright-free-stock-iconset.
html">iconeden</a></strong>
                    <p id="footerInfo">&copy; 2009  <a
href="http://rockablepress.com/">rockablepress</a></p>
            </div><!-- end footer -->
</div><!-- end footerWrap-->

</body>
</html>
```

The only unique section in our markup, when compared to the home page, is our form. As we've yet to provide styling for form elements, the page currently looks like so:



We'll have to remedy this!

So within your stylesheet, add the following:

```
/******** FORMS **********************/

form {
    width: 464px;
    overflow: hidden; }

label {
    display: block;
    font-size: 11px;
    color: #645a56; }

input[type=text],
textarea {
    width: 452px;
    height: 27px;
    border: 1px solid #cec5bf;
      margin-bottom: 27px;
    color: #645a56;
    font-size: 17px;
    padding: 7px 0 0 10px;
    font-family: helvetica, arial; }

input[type=text]:hover, textarea:hover, input:focus,
textarea:focus {
    background: #eeecec;
    outline: none; }

#container textarea {
    height: 212px; }
input[type=submit] {
    border: 1px solid #aba199;
    text-align: center;
    color: #30241f;
    font-size: 11px;
```

```
        width: 100px;
        padding: 12px;
        text-transform: uppercase;
        float: right;
        margin-top: -12px;
        background: #d5ceca url(../images/submitButtonBG.jpg)
    repeat-x;
        cursor: pointer; }

    input[type=submit]:hover {
                background: #c8c0bd url(../images/
    submitButtonBG.jpg) repeat-x  -100px 0; }

    /************ END FORMS ************/
```

Based on my measurements from the “contact.psd” file, the values listed above seem to be accurate. Yours might be slightly different, and that’s perfectly okay.

You might be a bit confused when it comes to the input[type=text] syntax. In addition to targeting elements via ids and classes, we can also target some elements based upon their “type.” In these instances, I’ve selected all inputs whose “type” attribute are equal to “text”, or all standard textboxes. This selector will not target password inputs or text areas.

Note that this advanced syntax does not work in Internet Explorer 6 or below. However, we’ll fix this soon by importing a JavaScript file created by Dean Edwards (http://dean.edwards.name).

ROCK* TIP

Never allow browser deficiencies to deter you from advanced selectors. There are plenty of JavaScripts available that remedy IE6’s shortcomings.

We use this syntax again to target our “Submit” button, as well as when it’s

hovered over with the mouse. Of course, we could still use an id instead, but this is more fun!

```
input[type=submit] {
    border: 1px solid #aba199;
    text-align: center;
    color: #30241f;
    font-size: 11px;
    width: 100px;
    padding: 12px;
    text-transform: uppercase;
    float: right;
    margin-top: -12px;
    background: #d5ceca url(../images/submitButtonBG.jpg)
repeat-x;
    cursor: pointer; }

input[type=submit]:hover {
            background: #c8c0bd url(../images/
submitButtonBG.jpg) repeat-x  -100px 0; }
```

Cufón Font Replacement

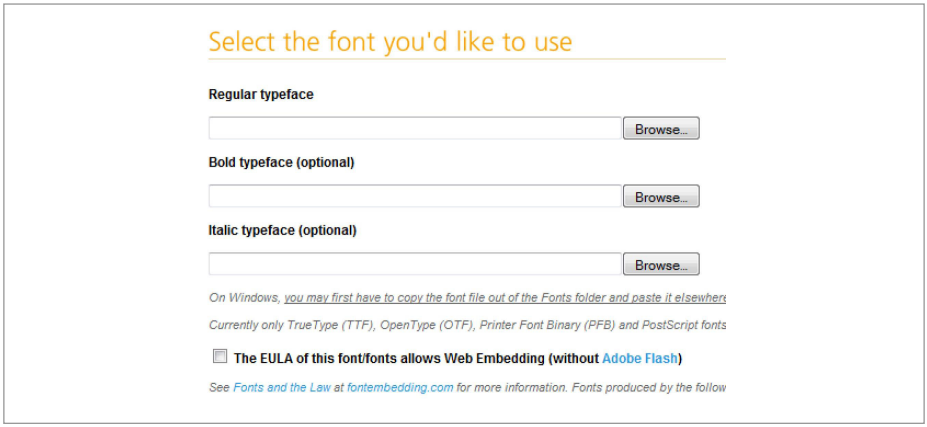
There are literally dozens of ways to use unique fonts in your projects. Having said that, they all share one thing in common: they're not perfect. Some tools implement Flash .swf files to get the job done, others use Canvas (HTML5 feature that allows for the rendering of bitmap images), and some even render text that isn't selectable. The bottom line is that you need to weigh your options and choose the easiest and best solution for the task at hand.

Considering how we're working on a relatively simple site, I've chosen to use Cufón (http://wiki.github.com/sorccu/cufon). If you've worked with sIFR, or Scalable Inman Flash Replacement (http://www.mikeindustries.com/blog/sifr/) before, you'll

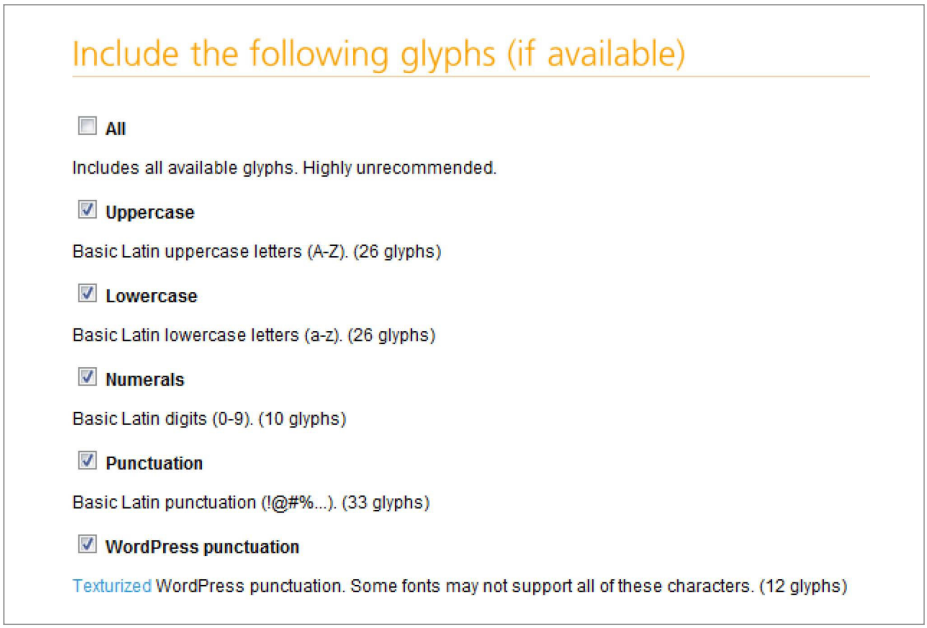
undoubtedly have found yourself thumping your skull at some point or another. Luckily, with Cufón, it just works! Let's see how.

1. Visit Cufón's website (cufon.shoqolate.com) and right-click on the "Download" button at the top. Choose "Save-As" and place it on your desktop.

2. In order to function, we need to use the font converter utility on Cufón's website. Refer to this page: http://cufon.shoqolate.com/generate/.



3. Find your desired font. In our case, it's located within *Final Project* > *Assets* > *Fonts* > *duepuntozero.ttf*.



4. Include the uppercase, lowercase, numerals, and punctuation glyphs only. The others aren't necessary and will only increase your file size.

5. Cufón allows you to designate a specific URL for your file, to increase security. It's extremely important that you ensure that you have the proper privileges to use a font. The font we're using is free. Type your site's URL into the "Security" textbox.

Since we're just getting started, you can leave the final two sections at their default values. Accept the terms, and click "Let's Do This". You'll then be presented with a download box asking you where to save the generated script. Once again, save it to your desktop for easy retrieval.

6. Create a new folder in your project and name it “js”.

7. Place the Cufon.yui.js and your newly created Duepuntozero_400.font.js files.

8. Within the head tag of each page, append the following code:

```
<script type="text/javascript" src="js/cufon-yui.js"></
script>
<script type="text/javascript" src="js/Duepuntozero_400.
font.js"></script>
<script type="text/javascript">
  Cufon.replace('h2');
</script>
```

Believe it or not, that’s it! By calling all “h2” tags within Cufon.replace, we’re telling the engine to find all heading two tags on the page and replace them with our new font! The only downside is that the final rendered text is not selectable.

Truthfully, we could have used this same method for our subheadings, like “Web Development, Graphics”, etc, but I wanted to show you how to use CSS sprites! Either method would work just fine. There are merits to both.

4

Compensating for Older Browsers

There will be a day when web designers can joyfully ignore ancient browsers like Internet Explorer 6, and hopefully 7! Unfortunately, we're not quite at that point yet. In fact, at the time of writing this, according to w3schools.com *(http://www.w3schools.com/browsers/browsers_stats.asp)*, 13.6% of all users are still browsing with this decade-old browser.

2009	IE7	IE6	IE8	Firefox	Chrome	Safari	Opera
August	15.1%	13.6%	10.6%	47.4%	7.0%	3.3%	2.1%
July	15.9%	14.4%	9.1%	47.9%	6.5%	3.3%	2.1%
June	18.7%	14.9%	7.1%	47.3%	6.0%	3.1%	2.1%
May	21.3%	14.5%	5.2%	47.7%	5.5%	3.0%	2.2%
April	23.2%	15.4%	3.5%	47.1%	4.9%	3.0%	2.2%
March	24.9%	17.0%	1.4%	46.5%	4.2%	3.1%	2.3%

Cringe Time

Viewing your website in Internet Explorer 7 and below for the first time will make you cringe. You might get lucky, but most of the time, you'll find numerous issues that need to be fixed.

So let's jump in and see how much more work we'll have to perform.

Tools

To view your site in IE6 – IE8, you have a few options.

1. On the PC, download IE Tester. This program will allow you to view your site in every IE browser. http://www.my-debugbar.com/wiki/IETester/HomePage.

2. There are many online sites that will take snapshots of your site in dozens of browsers. My personal favorite is http://www.browsershots.org. For these to work, the site needs to be online – not on your development server.

For this book, we'll be utilizing the first option. If, however, you're a Mac user (such as myself), feel free to use browsershots.org, or set up a virtual machine to run Windows. When testing in IE6 on the Mac, you'll find that your best option is to either temporarily hi-jack a PC, or use a program such as Vmware Fusion (http://www.vmware.com/products/fusion/).

Yikes! Truth be told, this isn't too bad when compared to some sites. Mostly, we need to fix some spacing issues. Internet Explorer 6 is ten years old. The fact that we're compensating for it at all is nice enough! No migraines necessary.

While these inconsistencies might be intimidating for beginners, you'll quickly learn that a few extra properties can fix huge layout issues. The majority of these issues can be fixed by giving an element "layout."

> *"Layout" is an IE/Win proprietary concept that determines how elements draw and bound their content, interact with and relate to other elements, and react on and transmit application/user events."*
>
> *On Having Layout http://www.satzansatz.de/cssd/onhavinglayout.html.*

Why is "Having Layout" Necessary?

It's important because it forces the element to have a rectangular shape. This means that the content of the element cannot flow around other boxes, which will often cause major layout issues with your website.

When Microsoft's proprietary property "hasLayout" is equal to true, the element "has layout." Luckily, we can trigger this by adding specific properties to our selectors. Personally, my favorite method is to set the height of an element to 100%. Other possibilities include specifying a width and height, creating a positioning context, and setting the "zoom" property equal to 1.

Targeting Specific Versions of Internet Explorer

To target a specific version of IE, we use conditional statements within the head tags of our HTML document. Add the following code to each page:

```
<!--[if lt IE 8]>

    <script src="http://ie7-js.googlecode.com/svn/version/
2.0(beta3)/IE8.js" type="text/javascript"></script>

<![endif]-->

<!--[if lte IE 6]>

    <link rel="stylesheet" type="text/css" href="css/ie6.
css" />
<![endif]-->
```

The first block states, “if the viewer is using anything less than Internet Explorer 8, then import the following script, otherwise do nothing.” We’re linking to a popular script called IE8.js, created by Dean Edwards (http://code.google.com/p/ie7-js/). This script allows us to use many advanced selectors in our stylesheets without having to worry about IE6 not recognizing them. For more details, visit http://code.google.com/p/ie7-js/.

The second block is specifically for IE6 users. Notice the first line: “if lte IE 6”. This means, “if the user is browsing with IE6 or lower, import the contained stylesheet.” Within these tags, we link to ie6.css. Go ahead and create that file now, and save it in your CSS folder. **Now, we can rest assured that any code within this file will only run if the user is browsing with IE6 or lower.**

Within this file, paste in the following code:

```
#container #nav li a {
float: left;
}

#container ul#nav, #slogan {
 overflow: hidden;
}

.arrow {
 display: none;
}

#secondaryContent cite {
 margin-top: -25px;
}

#recentWork ol {
 padding-right: 36px;
}


#main {
 height: 100%;
}
```

I'll be completely honest with you. Sometimes, we just have to accept that IE6 is really strange. When we encounter a weird positioning quirk, we have two options:

1. Spend hours calculating exactly how and why this issue occurred.

2. Save yourself the headache and "wing it." If you can't figure out why IE6 needs that extra 27px worth of

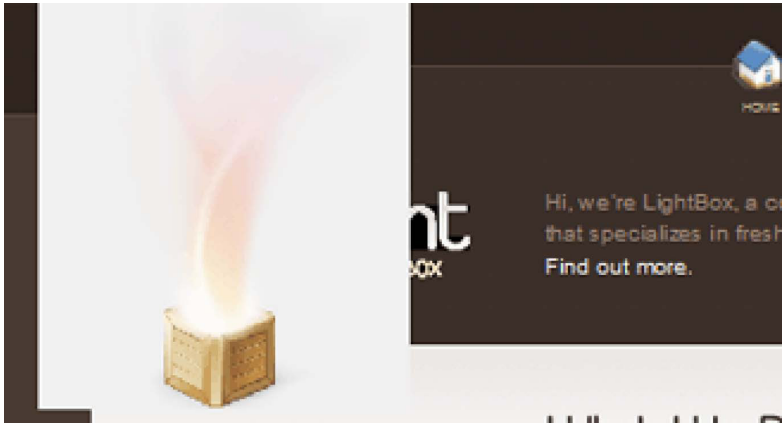
padding, don't worry about it! Just compensate and move on. You could easily waste hours upon hours trying to understand IE6, but ultimately, you'll be learning about a browser that is slowly fading from use. It's better to concentrate on standards-based browsers.

ROCK* TIP

You'll learn to accept that, sometimes, seemingly random properties are required to make IE6 happy!

PNG-24 Transparency in IE6

Notice that nasty gray background behind our Light Box? This is occurring because IE6 does not natively support the PNG-24 format. As a result, rather than displaying transparency, it instead renders an ugly gray background.



Luckily, there are a handful of solutions:

Unit PNG Fix

- **Download**: http://labs.unitinteractive.com/unitpngfix.php.
- **Issues**: It doesn't correctly tile transparent background images. Instead, it'll stretch your image. Not a huge issue, but beware. Other than that quirk, this method works perfectly. I find myself using it more than the others.

DD_BelatedPNG Fix

- **Download**: http://www.dillerdesign.com/experiment/DD_belatedPNG/.
- **Issues**: Every fix has a few quirks, but this one works quite well, other than the fact that it can be a bit tricky to set up.

IE8.js Fix

- **Download**: http://code.google.com/p/ie7-js/.
- **Issues**: You'll find that this file is larger than the others. That is because fixing the transparency issues is only part of what it does! It additionally brings many other IE6 deficiencies up to modern standards. Keep this in mind when choosing. If you only want to show transparency, this might not be the best choice.

Twin Helix Fix (from Angus Turnbull)

- **Download**: http://www.twinhelix.com/css/iepngfix/.
- **Issues**: Angus has updated this file in the last six months – it now properly implements background-repeat/position! Rather than referencing a Javascript file, this fix requires the use of the "behavior" CSS property.

We'll be using that final option: "Twin Helix Fix." It's a simple matter of downloading the "iepngfix.htc" file from the link above, and dragging it into your project. Next, we reference the file by using the "behavior" property.

We need to fix two images: the Light Box, and the logo within the footer.

Append the following to your IE6 stylesheet:

```
#footer img, #slogan img {

    behavior: url(../iepngfix.htc);

}
```

Now save the page, and refresh your browser. Easy!

Note that you can additionally use the IE8.js script that we previously imported to accomplish this. However, you must change the name of all images that make use of transparency. Simply append “-trans” to the end of the file name, and the rest will be done for you!

Use the method that you feel most comfortable with.

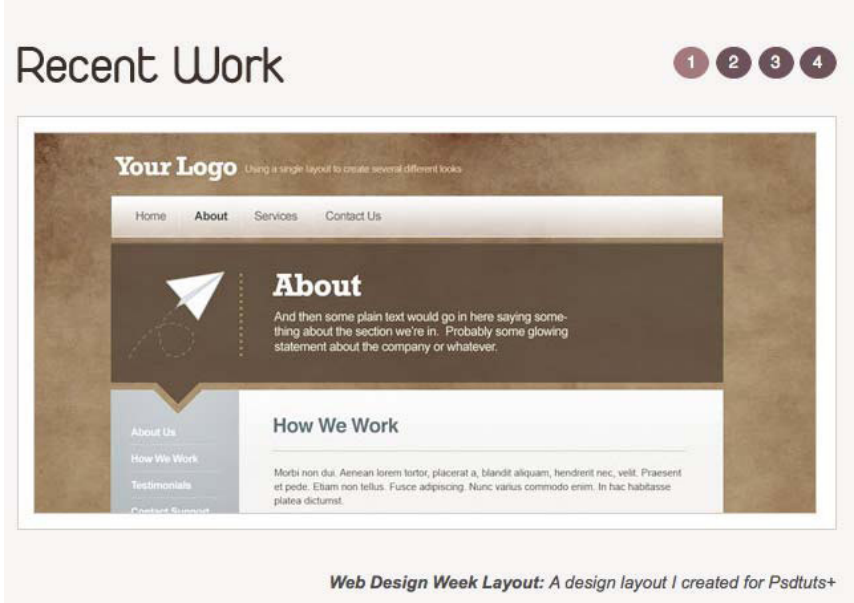
5

Bonus Chapter: A Splash of jQuery

Earlier in the book, I promised that I'd show you how to create the "transition" effect on the Home Page.

Using jQuery, such tasks become trivial. Because we aren't using too much Javascript in this project, you can keep it inline, if you like. However, in the source code, I will move this script into its own external file. Let's begin.

The Script



Web Design Week Layout: *A design layout I created for Psdtuts+*

1. Just before the closing <body> tag, reference Google's CDN to import the jQuery library.

```
<script src="http://ajax.googleapis.com/ajax/libs/
jquery/1.4/jquery.min.js" type="text/javascript"
charset="utf-8"></script></body>
```

2. Within script tags, append the following code:

```
$('#recentWork li a').click(function() {

    var $this = $(this);

    if ( !$this.parent('li').hasClass('chosen') ) {
```

```
            var url = this.href,
                  height = $('.image_display img').
css('height');

            $this
              .parents('ol')
              .find('li.chosen')
                  .removeClass('chosen');

            $this.parent('li').addClass('chosen');

            $('.image_display')
              .css('height', height)
              .children('img')
                  .fadeOut(400, function() {
                        $(this).attr('src', url).
load(function() {
                              $(this).fadeIn(400);
                        });

                  $this.parents('#recentWork')
                    .find('p:last')
                        .empty()
                        .html('<p>' + $this.attr('title')
+ '</p>');
            });

    }

    return false;

});
```

Decoding the Script

```
$('#recentWork li a').click(function() {
```

Find the anchor tag within the recentWork div and listen for when the user clicks it.

```
[replace with]

var $this = $(this);

if ( !$this.parent('li').hasClass('chosen') ) {

    var url = this.href,
            height = $('.image_display img').css('height');

    $this
      .parents('ol')
      .find('li.chosen')
            .removeClass('chosen');

    $this.parent('li').addClass('chosen');
```

Create a new variable called "$this", and make it equal to the anchor tag that was clicked on. This will "cache" the location of "this" so that it doesn't have to search the DOM every time it's referenced.

If the anchor tag's parent – the list item – does not have a class of "chosen", add one, and make sure that no other list item has this same class. Adding this class will allow us to change the background color of the selected item. All we need to do is return to our stylesheet and add an additional style.

```
#recentWork ol li.chosen a, .ourWorkol li.chosen a {
    background: #a37a7e;
}
```


```
$this
  .parents('ol')
  .find('li.chosen')
    .removeClass('chosen');

$this.parent('li').addClass('chosen');

$('.image_display')
  .css('height', height)
  .children('img')
    .fadeOut(400, function() {
            $(this).attr('src', url).load(function() {
                    $(this).fadeIn(400);
            });

    $this.parents('#recentWork')
      .find('p:last')
            .empty()
            .html('<p>' + $this.attr('title') + '</p>');
});

}

return false;
```

Get the wrapping div with a class of “image_display” and adjust its CSS to fix a weird IE6 quirk.

Next, fade out the main image, change its source to the url of the anchor tag that was clicked, and then load this new image. Once it’s finished loading, fade the image back in over the course of a half-second. Finally, empty the current text from the description and replace it with whatever was in the “title” attribute of the anchor that was originally clicked on.

Lastly, “return false” to disable the anchor tag’s default action.

So that’s it folks! You’re done!

Conclusion

So you've done it! After 100+ pages, we've battled the forces of evil and wrangled our first design into a compliant HTML and CSS website. However, there's only so much that can be taught within the confines of a book, so I encourage you to continue learning in any way that you can. Live at the bookstore, read blogs, and write tutorials. The sky's the limit!

Thanks for reading!

Appendix

Further Study

Articles

How to Design and Code a Flexible Website – http://net.tutsplus.com/tutorials/html-css-techniques/how-to-design-and-code-a-flexible-website/

From PSD to HTML, Build a Set of Website Designs Step by Step – http://net.tutsplus.com/tutorials/site-builds/from-psd-to-html-building-a-set-of-website-designs-step-by-step/

Design and Code a Slick Website From Scratch

Part 1 – http://net.tutsplus.com/tutorials/design-tutorials/design-and-code-a-slick-website-from-scratch-%e2%80%93-part-i/

Part 2 – http://net.tutsplus.com/tutorials/html-css-techniques/design-and-code-a-slick-website-from-scratch-%e2%80%93-part-ii/

Build a Sleek Portfolio Site from Scratch – http://net.tutsplus.com/tutorials/site-builds/build-a-sleek-portfolio-site-from-scratch/

Video Tutorials

Slice and Dice That PSD – http://net.tutsplus.com/videos/screencasts/slice-and-dice-that-psd/

Converting a Design From PSD to HTML – http://net.tutsplus.com/videos/screencasts/converting-a-design-from-psd-to-html/

How to Convert a PSD to HTML – http://net.tutsplus.com/videos/screencasts/how-to-convert-a-psd-to-xhtml/

About The Author

Jeffrey Way is part of the Envato team. He is the Editor of Nettuts+, a web development tutorials blog with over 50,000 daily readers, and he's also the manager of two marketplaces for web developers: ThemeForest and CodeCanyon. He has been in the web industry for over 5 years, with expertise in HTML, CSS, PHP, JavaScript, jQuery, CodeIgniter, Database Development and WordPress.

Jeffrey lives in Tennessee, USA, with his fiancé Allie and their little dachshund. "I spend too much time in front of the computer" he says, "and I find myself telling my fiancé, "We'll go in 5 minutes!" far too often. I just can't go out to dinner while I'm still producing FireBug errors... it drives me crazy."

In his free time, among other things – like writing this book – he writes articles for his own personal blog (www.Jeffrey-way.com).

In Photoshop to HTML, Nettuts+ editor and author Jeffrey Way will take you through the entire process of converting a design from Photoshop into a complete HTML/CSS website!

This book also comes with the completed demo site which you're free to use as you wish in your own projects. But not only that, you also get an additional series of screencasts where Jeffrey Way covers the entire project from start to finish. So you have the choice of either reading or viewing!

So what are you waiting for?! Start slicing!

net tuts+

ROCKABLE