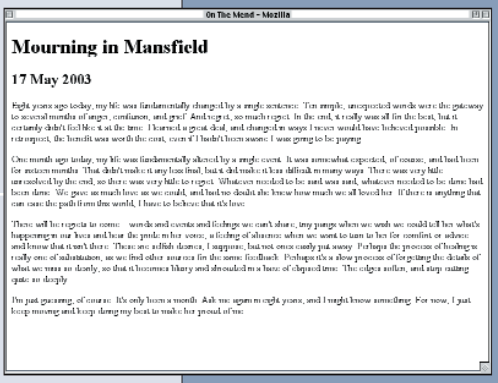
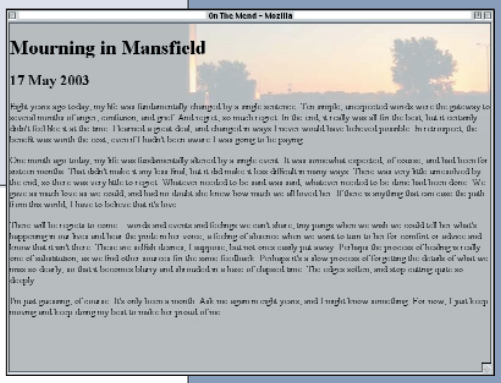
4

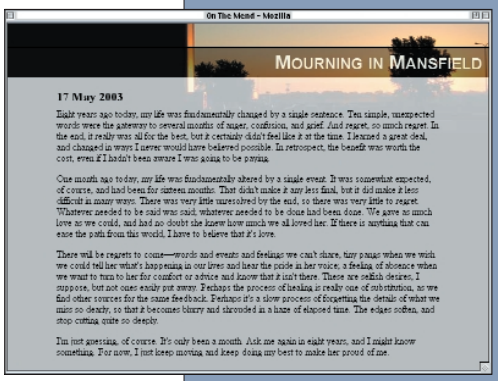
Positioning in the Background

But I was told there would be no math!
Too bad.

—*Macintosh Technical Note #31*







It's a common thing, at least in print design, to use shaded variations of a background to make portions of the design stand out. A good example is an ad in which there's a big picture of a mountain or beach or beautiful woman filling the entire ad, and in the middle is some compelling yet meaningless text, and surrounding that text is a region where the picture in the background has been washed out, as if the text were written on a half-opaque block of plastic.

Since opacity styles aren't part of CSS as of this writing, it's been generally thought that such effects are effectively impossible. There are fixed-attachment backgrounds (see Project 11 in *Eric Meyer on CSS* for more details), but they aren't supported by Explorer for Windows. One can use semi-opaque PNG graphics, but they aren't supported by Explorer for Windows. In fact, short of hacking the browser with proprietary behavior scripts, there's only one way to get a smooth translucency effect in Explorer for Windows, and that's by manipulating the position of background images.

Project Goals

We've taken on a project in which a local author is publishing some of his short essays and wants them to look artistic. He's a big fan of translucency effects, so he wants to see them used in his designs. Specifically:

- We are to use a sunrise picture for the first essay, "Mourning in Mansfield." This will include a dark shade over the background behind the title and a lightening effect over the background behind the essay's main text.
- For the essay "Gathering Stormclouds," we'll be using a picture of clouds at sunset. For this one, the title will have a lightening effect for the background, while the main text will have a darkened background and light text.

The author is supplying the images, so fortunately we don't have to worry about acquiring them. All we really have to do is pull a little sleight of style to get the translucency effects our client has requested.

Preparation

See the Introduction for instructions on how to download files from the Web site.

Download the files for Project 4 from this book's Web site. If you're planning to play along at home, load the file `ch04proj.html` into the editing program of your choice. This is the file you'll be editing, saving, and reloading as the project progresses.

Style at Dawn

For the first half of this project, we'll take the first of the two documents and add the styles necessary to create translucent effects using ordinary JPEG images. As we'll soon see, the images in question make it fairly easy to create an attractive design.

Getting Started

As usual, our first step should be to look at the structure of the document. As we can see from Listing 4.1, there isn't a whole lot to it really, just a masthead `div` with a heading and a "main" `div` containing the text of the entry. Without any styles, we get the very plain rendering shown in Figure 4.1.

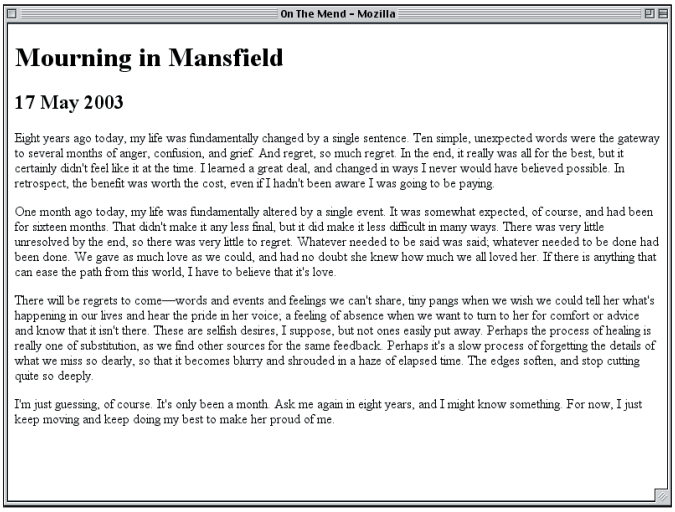


FIGURE 4.1

The journal entry in its unstyled state.

Listing 4.1 The Basic Document Structure

```
<div id="masthead">
  <h1>Mourning in Mansfield</h1>
</div>
<div id="main">
  <h2>17 May 2003</h2>
  [...entry text...]
</div>
```

Now, with nothing more than what we already have in the document and some background images, we need to create a tasteful design that makes use of translucent effects. To aid us in this goal, we have the three images shown in Figure 4.2: a basic image (`morn-base.jpg`), a faded version of that same image (`morn-fade.jpg`), and a washed-out version (`morn-wash.jpg`).

FIGURE 4.2

The three background images we have at our disposal.

To lay the groundwork, the first thing we want to do is strip the "gutter" from the `body` element itself. We'll do that with a familiar rule:

```
<style type="text/css">
body {margin: 0; padding: 0;}
</style>
```

The next thing to do is drop an image into the `body`'s background. It's important to choose the right one since the background color will have to match the background image, and the body background is the one that will be seen throughout most of the design. For maximum readability, we'll pick the washed-out version of the image for the body. A quick check with a color-sampling tool reveals that the background value is a shade of gray that's 71% white, so we'll do the following:

```
body {margin: 0; padding: 0;
  background: rgb(71%,71%,71%) url(morn-wash.jpg);}
```

Words Instead of Numbers

We could have used the keywords `right top` instead of the values `100% 0` for the background position, but since later backgrounds will be positioned using offsets, we're going to avoid the use of keywords throughout this project.

That's a start, but we'll need more. As this rule stands, the image will start out in the upper-left corner of the body's background and will tile infinitely both horizontally and vertically. For our design, we want the image to appear once and *not* repeat, and we want to place it in the top-right corner. We can do that by adding just a bit more to the background declaration, with the result shown in Figure 4.3.

```
body {margin: 0; padding: 0;
  background: rgb(71%,71%,71%) url(morn-wash.jpg) 100% 0 no-repeat;}
```

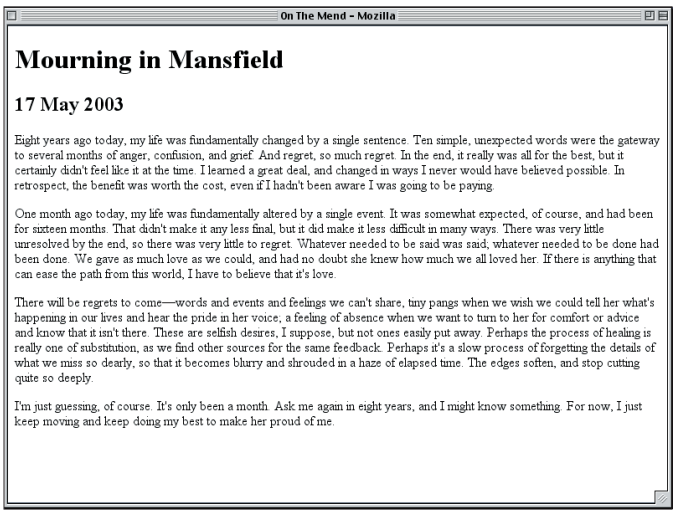


FIGURE 4.3

Starting out with some basic styles for the `body` *element.*

Masthead Styles

The document's masthead is next to fall under our scrutiny. We'll apply the basic background image (`morn-base.jpg`) to the masthead itself, but let's consider for a moment exactly how.

Our aim is to have the background images line up with each other to create the illusion of translucency. To make that happen, we'll need to make sure the masthead's top edge is lined up with the body's top edge. We'll also need to get the background image to not repeat and to sit in exactly the same place as the body's. Thus, the values `100% 0` and `no-repeat` are going to make another appearance. A quick run through a color sampler gives us the background color we need to match the image.

```
body {margin: 0; padding: 0;
  background: rgb(71%,71%,71%) url(morn-wash.jpg) 100% 0 no-repeat;}
#masthead {background: rgb(2%,4%,4%) url(morn-base.jpg)
 100% 0 no-repeat;}
</style>
```

There is one more thing that needs to be done: We need to take the margins off the `h1` element inside the masthead `div`. If we don't, the margins will actually stick out of the `div` in CSS-conformant browsers, which will push the top edge of the `div` down from the top of the document. By removing the margins, we get the result shown in Figure 4.4.

```
#masthead {background: rgb(2%,4%,4%) url(morn-base.jpg)
  100% 0 no-repeat;}
#masthead h1 {margin: 0;}
</style>
```

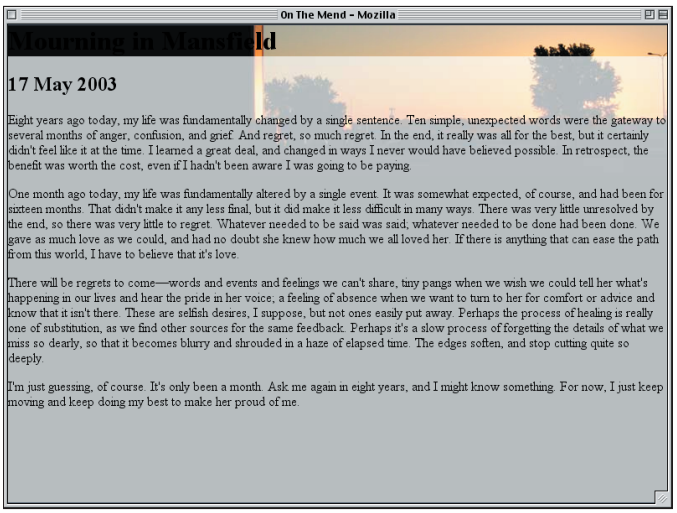


FIGURE 4.4

A different background image is added to the masthead.

Thanks to the masthead and body lining up along the top edge, their backgrounds do the same thing, which gives us exactly the kind of effect we wanted. Although the masthead's background is actually "on top" of the body's, it looks as if there is a grayish translucent layer between us and the "page background." It's an illusion, but a very useful one!

Of course, we can't leave things as they are. The heading text is now basically black on a black background, so it isn't very readable. Let's change its color to a light color that reflects the "sunrise" theme and update its font styling to look a little more professional.

```
#masthead h1 {margin: 0;
  color: #EED;
  font: small-caps bold 2em/1em Arial, sans-serif;}
</style>
```

Default Line Heights

Why bother with the explicit `1em` value for `line-height`? Because that isn't the default—most browsers default to a `line-height` value somewhere around 1.2. That makes the height of each line a little taller than the `font-size` value. Our `1em` value overrides that default behavior, which will come in handy later on.

The `font-size` and `line-height` values were actually chosen with some care. By setting the `font-size` of the `h1` to `2em`, we've made the text twice as big as its parent element, the `#masthead div`, which inherits its `font-size` value from its parent, the `body`. With the `line-height` value of `1em`, we've defined the height of the `h1`'s content to be exactly equal to its `font-size`.

Actually, now that we've cleaned up the masthead text, it might be nice to increase the height of the masthead by dropping some top padding on the masthead `div`.

```
#masthead {padding: 2.5em 0 0;
  background: rgb(2%,4%,4%) url(morn-base.jpg) 100% 0 no-repeat;}
```

By doing this, we're basically wedging a space between the top border of the masthead `div` and the top margin edge of the `h1`. By making the top padding value `2.5em`, we allow the spacing to stay in proportion to any changes in text size, no matter how they happen.

With this done, we can see a lot of improvement in Figure 4.5, although we aren't home free just yet.

The light text is now on a background that's mixed light and dark, which will make the text very difficult to read no matter what color we assign. We could put the text over to the right side so that it's more over the light portion of the background, but the dark tree will still prevent us from changing the text to be dark.

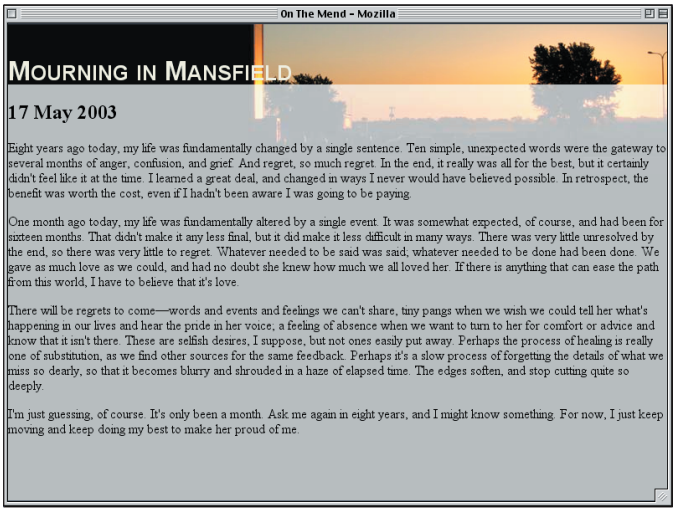


FIGURE 4.5

Padding the masthead and styling the heading bring drastic improvements.

What we need is a background for the heading itself that's more consistent, and fortunately, we already have one: `morn-fade.jpg`. We can add that to the background of the `h1` element.

```
#masthead h1 {margin: 0;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% 0 no-repeat;
  color: #EED;
  font: small-caps bold 2em/1em Arial, sans-serif;}
```

Since we now have a darker background for our light text, we can actually shift it over to the right side of the design. At the same time, we'll pad out the `h1` element a little bit so that the text doesn't get too close to the edge of the dark area.

```
#masthead h1 {margin: 0; padding: 0.25em 0.33em;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% 0 no-repeat;
  color: #EED;
  font: small-caps bold 2em/1em Arial, sans-serif;
  text-align: right;}
```

The quarter-em top and bottom padding will nicely center the text in the dark background, and the third-em padding on the right and left will keep the title text from getting too cozy with the edge of the browser window, as illustrated in Figure 4.6.

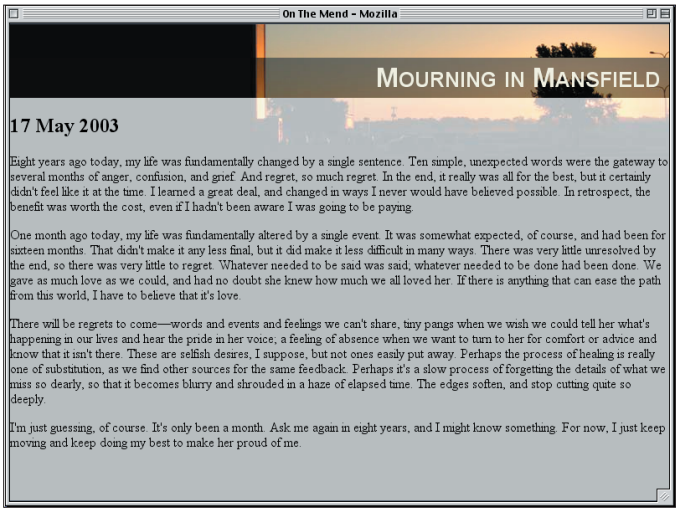


FIGURE 4.6

The third background is added, although it's out of alignment.

Whoops—there's a problem. The background image we just added isn't lining up with the other images, and this is creating a discontinuous tree. That's because the top edge of the `h1`'s background area isn't lined up with the tops of the other two.

We could fix this by moving the `h1` to the top of the page, but that wouldn't look very good. Instead, let's bring the `h1`'s background into alignment with the others but leave the element where it is now. To pull this off, we'll have to do a little math.

Measuring Ems

Remember that one em is always equal to the `font-size` of the element on which the em is being used. Thus, given our styles, `1em` for the `h1` is twice the size of `1em` for a paragraph.

We know that the `h1`'s top edge is 2.5em from the top of the masthead, thanks to the top padding we gave the masthead `div`. However, the `h1`'s `font-size` is twice that of the masthead `div`'s, due to the `2em` value in the `font` declaration, so we have to divide that number in half to get the right offset. Therefore, if we position the background image of the `h1` so that its top edge is actually –1.25em above the top edge of the `h1`'s background area, then all the backgrounds should line up.

```
#masthead h1 {margin: 0; padding: 0.25em 0.33em;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% -1.25em no-repeat;
  color: #EED;
  font: small-caps bold 2em/1em Arial, sans-serif;
  text-align: right;}
```

One more touch should make the masthead look more polished, and that's a solid black border along the top and bottom of the dark background. However, doing this means that the background area of the `h1` will be effectively shifted downward by a pixel (thanks to the top border), which will cause the images to go out of alignment in CSS-conformant browsers. We can fix that with a negative one-pixel top margin.

```
#masthead h1 {margin: -1px 0 0; padding: 0.25em 0.33em;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% -1.25em no-repeat;
  color: #EED; border: 1px solid black; border-width: 1px 0;
  font: small-caps bold 2em/1em Arial, sans-serif;
  text-align: right;}
```

There's just one problem: IE/Win goes completely bonkers when the negative margin is introduced. So we'll actually need to change the margin back to what it was, and use another of IE's bugs against it to hide the negative margin.

```
#masthead h1 {margin: 0; padding: 0.25em 0.33em;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% -1.25em no-repeat;
  color: #EED; border: 1px solid black; border-width: 1px 0;
  font: small-caps bold 2em/1em Arial, sans-serif;
  text-align: right;}
html>body #masthead h1 {margin: -1px 0 0;}
</style>
```

Thanks to the `html>body` part of that rule, Explorer will just skip right over the new rule. Other browsers, such as Safari, Opera, and Gecko-based browsers like Mozilla, will understand and apply the rule. Thus, we get a good cross-browser layout.

Although these new styles may be off by a pixel in some browsers, it's an acceptable price to pay. The difference will hardly be noticeable one way or the other, as Figure 4.7 illustrates.

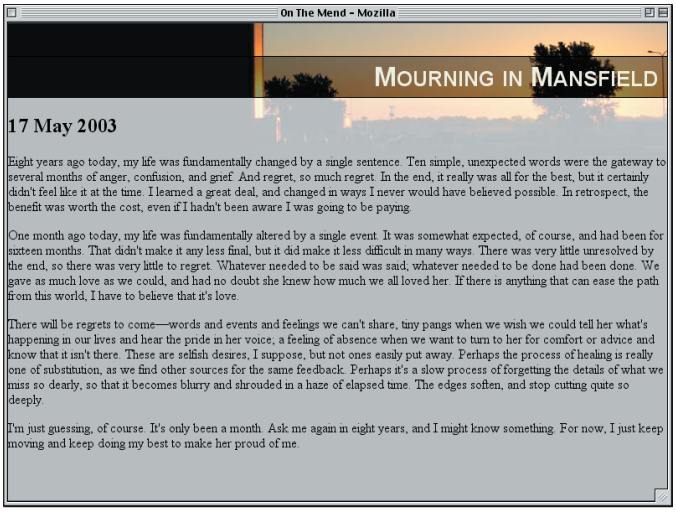


FIGURE 4.7

With a little negative positioning, the backgrounds are lined up nicely.

Cleaning Up

At this stage, the masthead is pretty well in hand; all that remains is to clean up the entry itself. Since the text doesn't have much to do with our background-alignment project, we'll just drop in all the styles at once. With some margins on the main `div`, no margins and a `font-size` for the `h2`, and some massaged margins for the paragraphs, we should be all set.

```
html>body #masthead h1 {margin: -1px 0 0;}
#main {margin: 1.25em 5em 0 5em;}
#main h2 {margin: 0; font-size: 1.25em;}
#main p {margin: 0.5em 0 1em;}
</style>
```

Thus, we arrive at the complete set of styles provided in Listing 4.2 and illustrated by Figure 4.8.

Listing 4.2 The Full Style Sheet

```
body {margin: 0; padding: 0;
  background: rgb(71%,71%,71%) url(morn-wash.jpg) 100% 0 no-repeat;}
#masthead {padding: 2.5em 0 0;
  background: rgb(2%,4%,4%) url(morn-base.jpg) 100% 0 no-repeat;}
#masthead h1 {margin: 0; padding: 0.25em 0.33em;
  background: rgb(4%,4%,4%) url(morn-fade.jpg) 100% -1.25em no-repeat;
  color: #EED; border: 1px solid black; border-width: 1px 0;
  font: small-caps bold 2em/1em Arial, sans-serif;
  text-align: right;}
html>body #masthead h1 {margin: -1px 0 0;}
#main {margin: 1.25em 5em 0 5em;}
#main h2 {margin: 0; font-size: 1.25em;}
#main p {margin: 0.5em 0 1em;}
#masthead h1 is missing the margin declaration from above. Should read:
#masthead h1 {margin: -1px 0 0; (and so forth) - DS
```

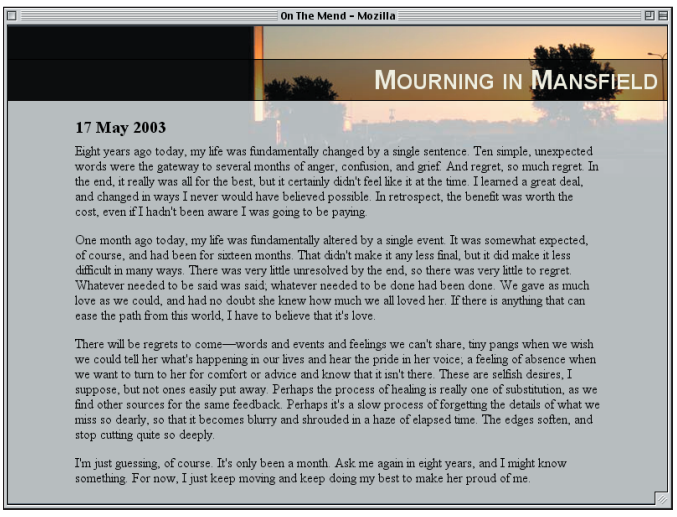


FIGURE 4.8
The completed journal styles.

To a degree, our job in this part of the project was easy because two of the three elements with backgrounds lined up along the top edge. Because that was the case, we could give two of the background images exactly the same position values, and the third only required a vertical offset.

Suppose, however, that we want to set up a translucency effect similar to those we've explored, but none of the element tops line up with each other. Given such a situation, we'd need to do a little more math and be careful about the length units we use. Let's move to the second document in our project and see how to handle such issues.

Beached Styles

In this half of the project, we'll take a new document with a slightly different structure and give it a translucent-effect makeover. We'll use the principles explored in the first half of the project, but we'll use them in a more sophisticated way because in this document none of the elements will line up along the top edge.

Assessing Structure and Style

As always, our first order of business is to scout out the lay of the land, so to speak, by examining the document structure and any styles that may already exist. The basic structure looks like this:

```
<div id="main">
  <h1>Gathering Stormclouds</h1>
  <div id="content">
    [...content...]
  </div>
</div>
```

As it happens, there's already a basic style sheet embedded with the document, and this leads to the result shown in Figure 4.9.

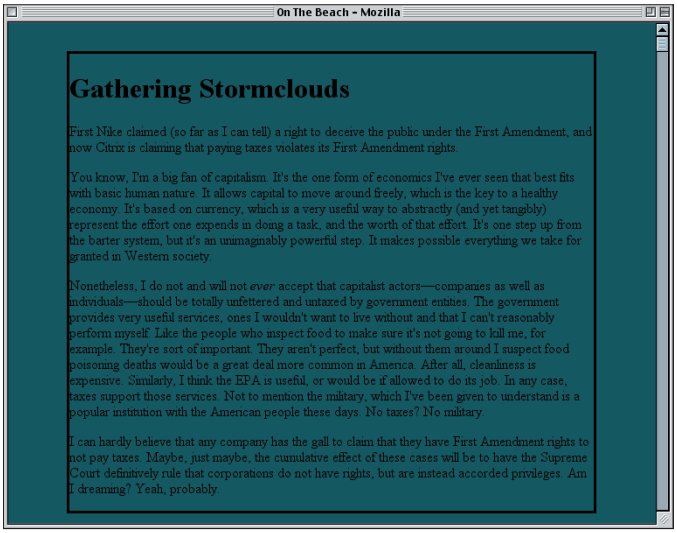
```
<style type="text/css">
body {margin: 0; padding: 0;
  background: rgb(14%,26%,30%);}
#main {width: 600px; margin: 2em auto;
  border: 3px solid black;}
</style>
```

Auto-Margins

By giving the main `div`'s left and right margins the value `auto` and the width an explicit value, we've centered the `div` within its parent. Note that this will not work in versions of IE/Win before IE6 that improperly centered elements using `text-align: center`.

FIGURE 4.9

Basic, if dark, beginnings for our new document.

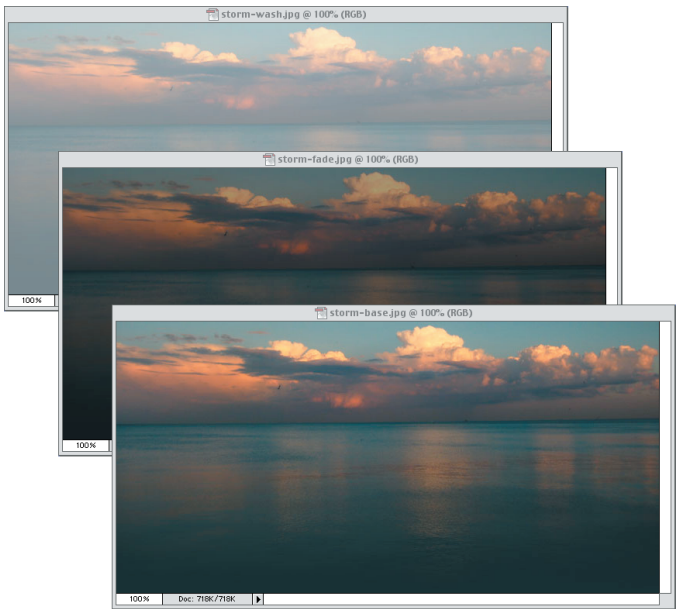


Of course, we'll be filling in other backgrounds, so the text will be more legible by the time we're done. We've just set up these styles to start so that we can dive into the background positioning.

Now we need to use each of the three images shown in Figure 4.10: a basic image (`storm-base.jpg`), a faded version of that same image (`storm-fade.jpg`), and a washed-out version (`storm-wash.jpg`).

FIGURE 4.10

The background images we have at our disposal.



For this design, we aren't going to apply any of these images to the `body` element. Instead, we'll use the main `div` and some of its descendants, but first let's style the text and place the elements.

Title Styles

The first thing we ought to do in setting up text and element styles is temporarily give the main `div` a background color. This will make it easier to see what we're doing.

```
#main {width: 600px; margin: 2em auto;
  border: 3px solid black;
  background: gray;}  /* temporary background */
```

Now we can get down to design. For the title of the page, we'll give it some size and height styles copied from the layout in the first half of the project.

```
#main {width: 600px; margin: 2em auto;
  border: 3px solid black;
  background: gray;}
#main h1 {font: 2em/1em "Times New Roman", serif;}
</style>
```

This will, as before, set the `h1`'s text in relation to its parent (in this case, `div#main`) and make sure its `line-height` is exactly equal to its `font-size`. We'll also center and lowercase the text and spread out the letters just a bit, as demonstrated by Figure 4.11.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;}
```

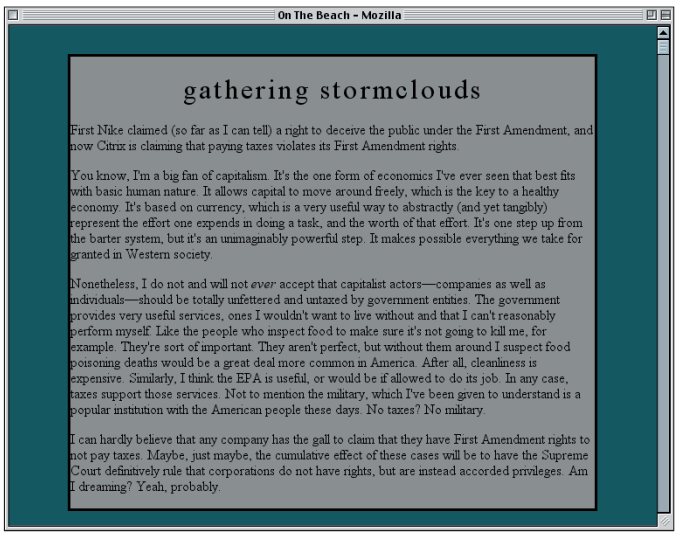


FIGURE 4.11

A temporary background helps us see the changes to the page title.

Quoted Fonts

It's only necessary to quote font names if they contain spaces or non-alphabetic characters.

Having the title all jammed up against the edges of the box feels a little claustrophobic, so let's add some margins to the `h1`.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0;}
```

To give some definition to the area of the design that holds the text, it might look kind of cool to add some thin borders around the title and the content. We'll start with the title but have borders appear only on the top, right, and left sides.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0;
  border: 1px solid black; border-bottom: none;}
```

Of course, this means that the title text is up against a border again, so we'll add in some padding.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0; padding: 0.5em 0.25em;
  border: 1px solid black; border-bottom: none;}
```

You may have noticed that we're sticking to margin and padding values that are evenly divisible by 0.25. This will make it much easier to do the math that's looming near the end of the project. In the meantime, let's see where we are now (see Figure 4.12).

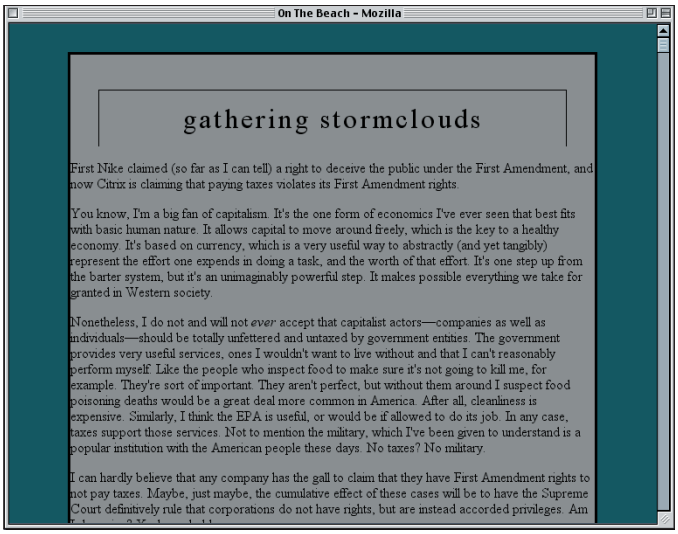


FIGURE 4.12

The addition of margins, padding, and a border makes the title stand out.

Content Styles

Bringing the main content of the page into line with the title shouldn't be too difficult. The first step is simple enough: We'll add a border to the content `div` that encloses all the text.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0; padding: 0.5em 0.25em;
  border: 1px solid black; border-bottom: none;}
#content {border: 1px solid black;}
</style>
```

That isn't sufficient, of course. If we left things there, the content `div` would be as wide as the main `div`, and the edges wouldn't line up with the edges of the title. Thus, we'll need to give the content `div` some margins. But how big?

Well, the `h1` was given a top margin 1.25em tall and side margins 1em wide. Recall, however, that the `font-size` value of the `h1` is `2em`. Since its em-based margins are calculated with respect to its calculated `font-size`, this means that we'll need to double the values for margins on our content `div`. That gives us 2em right and left margins and a 2.5em bottom margin. Since we want the content border right up against the title, that means no top margin.

```
#content {margin: 0 2em 2.5em;
  border: 1px solid black;}
```

As with the title, we don't want the content text getting too close to the borders around it. We could double the padding used on the title, but instead, let's give the content `div` a generous padding of an em and a half.

```
#content {margin: 0 2em 3em; padding: 1.5em;
  border: 1px solid black;}
```

Finally, let's add some styling to the content text. The first thing is to take the top margin off of all the paragraphs; this will ensure that the first paragraph is as close to the top of the content `div` as the `div`'s padding will allow while still keeping all paragraphs separated by an em. We'll also fully justify the text, as seen in Figure 4.13.

```
#content {margin: 0 2em 3em; padding: 1.5em;
  border: 1px solid black;}
#content p {margin: 0 0 1em; text-align: justify;}
</style>
```

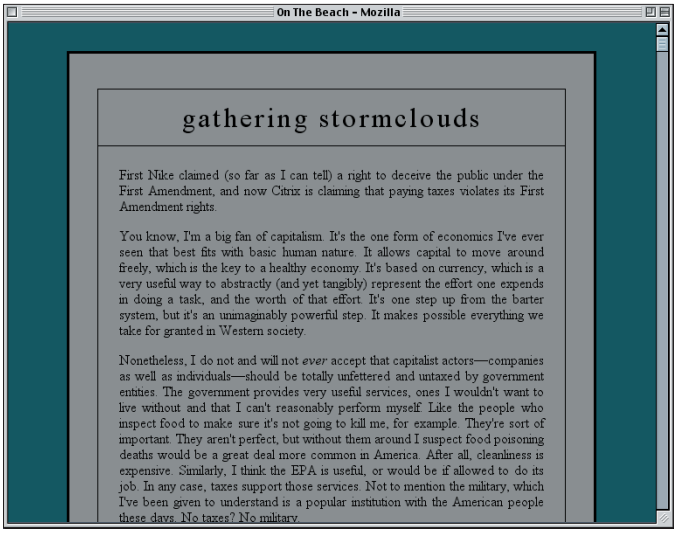


FIGURE 4.13
The main content integrates with the title.

At this point, we've done enough that we can start adding in the background images. So let's get started!

Adding the Backgrounds

Looking at the design in Figure 4.13 and the images we have available, let's take this approach: The main `div` will get the basic image (`storm-base.jpg`), the title will get the washed-out image, and the content `div` will get the faded image. This will necessitate changing the text color to something light, but we'll get to that in a minute. First, here's the main `div`'s background.

```
#main {width: 600px; margin: 2em auto;
  border: 3px solid black;
  background: rgb(7%,13%,15%) url(storm-base.jpg) 0 0 no-repeat;}
```

Next, we'll add a background to the title.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0; padding: 0.5em 0.25em;
  border: 1px solid black; border-bottom: none;
  background: url(storm-wash.jpg) 0 0 no-repeat;}
```

Finally, the content `div` gets its background.

```
#content {margin: 0 2em 2.5em; padding: 1.5em;
  border: 1px solid black;
  background: rgb(4%,8%,9%) url(storm-fade.jpg) 0 0 no-repeat;}
```

Alert readers will already have caught the problem: All of these elements set the background to start in their upper-left corner, but the elements don't line up. This means the backgrounds won't line up either, as we can see in Figure 4.14.

Top Left Zeroes

The background position value `0 0` is equivalent to `top left`. We're using numbers here because we're going to have to change some of them to negative values, and it isn't possible to combine keywords and numbers. Thus, something like `left 50px` is forbidden in CSS, but `0 50px` is not. (This restriction was removed in CSS2.1, but some browsers still obey the restriction.)

FIGURE 4.14

The backgrounds make their appearance, but they don't line up yet.

So now we have to figure out the offsets needed to get the background images to line up. The main `div` can be left alone, of course, so we'll work on the title first. It's actually pretty simple. We know that it has a top margin of 1.25em and a left margin of 1em, so those are the offsets we'll use because the em for margins is the same em that's used for background offsets.

```
#main h1 {font: 2em/1em "Times New Roman", serif; letter-spacing: 0.1em;
  text-transform: lowercase; text-align: center;
  margin: 1.25em 1em 0; padding: 0.5em 0.25em;
  border: 1px solid black; border-bottom: none;
  background: url(storm-wash.jpg) -1em -1.25em no-repeat;}
```

For the content `div`, the situation is a bit more complicated. The horizontal offset is easy: It's `2em`, the same as the left margin. For the vertical, though, we have to figure out how far it is from the top of the content `div` to the top of the main `div`.

Most of that distance is occupied by the `h1` element, so let's figure it out first. Within its own frame of reference, the content of the `h1` is effectively 1em tall, and it has top and bottom padding of 0.5em. It also has a top margin of 1.25em (and no bottom margin), and all that together adds up to 3.25em tall. We have to double that, remember, because of the doubled `font-size` of the `h1`. Therefore, in terms of the content `div`'s ems, the `h1` is 6.5em tall. With that information and a light red color in hand, we can update the rule appropriately.

```
#content {margin: 0 2em 2.5em; padding: 1.5em;
  border: 1px solid black;
  color: rgb(210,185,150);
  background: rgb(4%,8%,9%) url(storm-fade.jpg) -2em -6.5em no-repeat;}
```

Position and Order

We didn't get those backwards: `background-position` length and percentage values always put the horizontal offset before the vertical offset. Thus, `-1em -1.25em` offsets the image 1em to the left and 1.25em upwards. With keywords, the order is less important; `top left` and `left top` are equivalent, although the latter can trip bugs in the Netscape 6.x line.

That should do it—or should it? We've forgotten one thing: the top borders of the `h1` element and the content `div`, which cause a 2-pixel push downward for the background area of the content `div`. In fact, the `h1`'s background is also offset by a pixel, but we can let it slide since it isn't nearly as noticeable.

As it turns out, Explorer has forgotten those two pixels as well. Thanks to the way it places background images, the backgrounds already line up. So, if we were to change the offset to fix alignment in CSS-conforming browsers, the alignment would be thrown off in Explorer. So here's what we'll do: We'll leave the rule as it is and add in a new one that Explorer won't understand but more current browsers will.

```
#content {margin: 0 2em 2.5em; padding: 1.5em;
  border: 1px solid black;
  color: rgb(210,185,150);
  background: rgb(4%,8%,9%) url(storm-fade.jpg) -2em -6.5em no-repeat;}
html>body #content {margin-top: -2px;}
#content p {margin: 0 0 1em; text-align: justify;}
```

As in the previous half of the project, Explorer will just skip right over the rule we've just added. Other browsers, such as Safari, Opera, and Gecko-based browsers like Mozilla, will understand and apply the rule. Thus, we get a good cross-browser layout, as shown in Figure 4.15.

FIGURE 4.15

The backgrounds are now lined up quite nicely.

Pros and Cons

As powerful as this technique can be visually, it does make for pages that are weightier than they might otherwise be. In addition to the HTML and CSS, designs like this also require the loading of multiple backgrounds, most of which will only be seen in part. Thus, some of the bandwidth required for the page to download might be considered wasted since it contains image portions that the user will not see.

This is admittedly a drawback and one worth considering. In the examples used for this project, the background images were not too large, plus the faded and washed-out versions were highly compressed, which kept the weight down. A design that relied on five variations of a background image, with each one 800×600 or bigger, would create a monstrously weighty page!

Downloading full images and placing them as we did in this project carries a benefit: If the text in the document is resized, the translucency effects will still hold true (that is, unless text starts wrapping in unexpected ways). For example, in the "stormclouds" example, if the title text wrapped to two or more lines, the alignment of the content `div` would be thrown off quite a bit. We could in theory avoid the problem with fixed-attachment backgrounds, but Explorer for Windows doesn't support such techniques.

We could also get around the alignment problems caused by wrapping text with a simple change in structure. Instead of putting the content `div` after the `h1`, we could put the `h1` inside the content `div`. This, along with a suitable rearrangement of the CSS, would prevent misalignment if the title text ends up wrapping to multiple lines.

A much simpler and less weighty approach, at least in cases in which you want to lighten or darken a background, is to use semi-opaque PNG graphics for the elements that need to be faded or lightened. The problem, again, is Explorer for Windows, which supports PNG images but not the alpha channel, which wrecks the translucency. Instead, Explorer just shows a fully opaque version of the PNG. It's also the case that you can really only achieve lightening or darkening effects with PNGs, which limit the range of visual effects. With the offset-alignment approach we've discussed, you could use a basic image, one with the image contours traced, and another with a rippled-glass effect.

At any rate, it's important to employ the techniques explored in this project with care. They are most often useful in situations such as mastheads, section titles, or other places where the content is short and the area to be decorated is both wide and short.

Fixed-Background Layout

For more information on fixed-attachment layout techniques, see Project 12 of *Eric Meyer on CSS*.

Branching Out

Try creating styles to accomplish each of the variations described here. If the markup needs to be changed to make the variation easier or even possible, it will be noted in the text.

1. In the "Mansfield" design, try changing the backgrounds (and background colors) around. For example, you could use the washed-out background for the title, the faded image for the masthead, and the basic image for the body. Don't forget to change the colors, too!

2. In the "stormclouds" design, change the placement of the background images so that they're all centered horizontally within their elements. Make sure they all continue to line up with each other.

3. Change the "stormclouds" design as discussed in the "Pros and Cons" section, altering the markup slightly and rearranging the styles so that the visual outcome is the same but the layout is more resistant to text wrapping in the title.