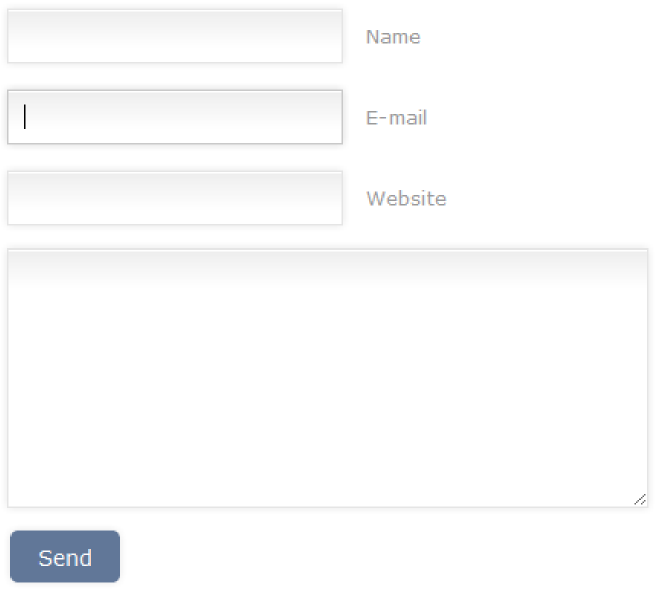
net.tutsplus.com

Design a Prettier Web Form with CSS 3

Thanks to advanced CSS properties, such as gradients and shadows, it's now quite easy to turn a dull web form into something beautiful – with minimal effort. I'll show you how in today's tutorial!

Our Final Product



Subtle background gradients give depth to the fields while shadows lift them from the page. Even more impressive is that this is done without any images at all.

By following this tutorial you will not only end up with a lightweight and beautiful form, you'll also learn and understand new CSS3 techniques, such as **box-shadow**, **gradients**, **opaque colors**, and **rounded corners**.

CSS3?

CSS3 is the next generation of CSS that is currently under development, but that doesn't stop browsers from already implementing many of the prominent features.

- Google Chrome (4.0)
- Mozilla Firefox (3.6)
- Safari (4.0)

Opera has greater levels of support for CSS3 (except background gradients) in their next version (10.50 Beta).

Internet Explorer has also stated that they will have improved CSS3 support with version 9; however, only time will tell how true this is.

The things you can do with CSS3 (shadows, gradients, round corners, animations, etc) all serve a purpose of creating beautiful effects without having to integrate images or scripts, resulting in quicker loading times.

Step 1: The HTML

Before we begin styling we need something to style, so here is the form.

```
<formclass="form">

  <pclass="name">
    <inputtype="text"name="name"id="name"/>
    <labelfor="name">Name</label>
  </p>

  <pclass="email">
    <inputtype="text"name="email"id="email"/>
    <labelfor="email">E-mail</label>
  </p>

  <pclass="web">
    <inputtype="text"name="web"id="web"/>
    <labelfor="web">Website</label>
  </p>

  <pclass="text">
    <textareaname="text"></textarea>
  </p>

  <pclass="submit">
    <inputtype="submit"value="Send"/>
  </p>

</form>
```

```
<form class="form">

    <p class="name">
        <input type="text" name="name" id="name" />
        <label for="name">Name</label>
    </p>

    <p class="email">
        <input type="text" name="email" id="email" />
        <label for="email">E-mail</label>
    </p>

    <p class="web">
        <input type="text" name="web" id="web" />
        <label for="web">Website</label>
    </p>

    <p class="text">
        <textarea name="text"></textarea>
    </p>

    <p class="submit">
        <input type="submit" value="Send" />
    </p>

</form>
```

Each field is inside a paragraph with its own class, and the three first fields have a label explaining their use.

How does it look without any styling?

Functional, but dull. Let's start pimping out this form.

Step 2: Basic Styling

Before we dive into the CSS3 techniques we need to create a basic layout for browsers that don't yet support CSS3.

```
input, textarea {
    padding: 9px;
    border: solid1px#E5E5E5;
    outline: 0;
    font: normal13px/100% Verdana, Tahoma, sans-serif;
    width: 200px;
    background: #FFFFFF;
    }

textarea {
    width: 400px;
    max-width: 400px;
    height: 150px;
    line-height: 150%;
    }

input:hover, textarea:hover,
input:focus, textarea:focus {
    border-color: #C9C9C9;
    }

.form label {
    margin-left: 10px;
    color: #999999;
    }

.submit input {
    width: auto;
    padding: 9px15px;
    background: #617798;
    border: 0;
    font-size: 14px;
    color: #FFFFFF;
    }
```

```
input, textarea {
    padding: 9px;
    border: solid 1px #E5E5E5;
    outline: 0;
    font: normal 13px/100% Verdana, Tahoma, sans-serif;
    width: 200px;
    background: #FFFFFF;
    }

textarea {
    width: 400px;
    max-width: 400px;
    height: 150px;
    line-height: 150%;
    }

input:hover, textarea:hover,
input:focus, textarea:focus {
    border-color: #C9C9C9;
    }

.form label {
    margin-left: 10px;
    color: #999999;
    }

.submit input {
    width: auto;
    padding: 9px 15px;
    background: #617798;
    border: 0;
    font-size: 14px;
    color: #FFFFFF;
    }
```

How does our effort look so far?

Not too bad. Now, let’s begin our enhancements with the more advanced CSS3.

Step 3: Box-shadow

Box-shadow does exactly what it sounds like: creates a shadow around a box.

The syntax for box-shadow is fairly simple:

1. box-shadow: <color> <horizontal offset> <vertical offset> <blur>;

```
box-shadow: <color> <horizontal offset> <vertical offset> <blur>;
```

Horizontal offset is the placement of the shadow from left to right. If you set it to “2px” the shadow will be 2 pixels to the right. **Vertical offset** is the same but up/down.

Blur is simply the amount of blur the shadow will have, where 0 is minimum.

This is how our box-shadow will look like:

```
input, textarea {
    box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    -moz-box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    -webkit-box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    }
```

```
input, textarea {
    box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    -moz-box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    -webkit-box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    }
```

Here we have three lines that look similar.

- **box-shadow** is pure CSS3 and so far only used in Opera.
- **-webkit-box-shadow** is for browsers using the Webkit engine, like Chrome and Safari.
- **-moz-box-shadow** is for browsers using Mozilla’s Gecko engine, like Firefox, Camino, Flock, and SeaMonkey.

Until CSS3 becomes the standard, you have to use all three methods. Internet Explorer has their own weird way of doing things, and although it’s capable of making a shadow it will not look the way we want it. 3

You might notice that there was no normal RGB color used, this is because we’re using two CSS3 techniques on the same line: **box-shadow** and **rgba**.

RGBA (Red Green Blue Alpha) is, simply put, color with opacity.

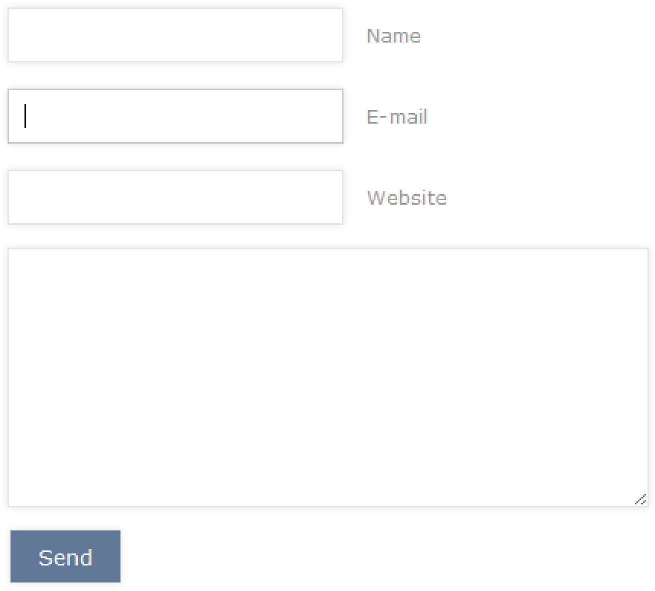
The syntax for rgba is this:

```
rgba(<red>,<green>,<blue>,<opacity>);
```

```
rgba(<red>,<green>,<blue>,<opacity>);
```

It’s perfectly fine to use a light grey for the shadow’s color, but if you are using any other background than white it will look strange. An opaque black on the other hand will work well no matter what background.

So our box-shadow is black with 10% (0.1) opacity, no horizontal and vertical offset, and with a blur of 8 pixels. It will look like this:



The keyword here is **subtlety**. If we apply too much shadow, it will look ugly; if we apply too little, it won't have an effect. Basically, we don't want anyone to notice the shadow, but still have it lift the fields from the page.

Step 4: Background Gradient

While the box-shadow syntax is easy to grasp, gradients are trickier. With CSS3 gradients, you can create some amazing shapes — from dart boards to rainbows — so as you can imagine it has a more complex syntax. Thankfully, we don't need to code a rainbow today; we just need a straight linear gradient.

Syntax for Webkit:

1. -webkit-gradient(linear, <start>, <end>, from(<color>), to(<color>))

```
-webkit-gradient( linear, <start>, <end>, from(<color>), to(<color>) )
```

Syntax for Gecko:

1. -moz-linear-gradient(<start> <angle>, <color>, <color>)

```
-moz-linear-gradient(<start> <angle>, <color>, <color>)
```

As you can see, the methods are quite different, so this will require some explaining.

Webkit gradients require a start point (X and Y), an end point (X and Y), a from-color, and a to-color. The angle is determined by where start and end are, and the gradient will be colored with the "from(color)" fading to "to(color)".

Gecko gradients, on the other hand, require only a start point (Y), and at least two colors. If you want a gradient going from top to bottom (90deg) you don't need to assign an angle.

So to get a simple linear gradient from top to bottom – black to white – we would do like this:

1. background: -webkit-gradient(linear, lefttop, leftbottombottom, from(#000000), to(#FFFFFF));

2. background: -moz-linear-gradient(top, #000000, #FFFFFF);

```
background: -webkit-gradient(linear, left top, left bottom, from(#000000), to(#FFFFFF));
background: -moz-linear-gradient(top, #000000, #FFFFFF);
```

And it would appear like this:

(I will continue to use the black color for demonstration; at the end, I'll switch to the real color we will be using for the form.)

Now that we have the basics out of the way, we can start making the form look how we want. The first thing we want to do is limit the height of the gradient so that it looks the same for both input fields and textarea; otherwise the gradient would fill the entire height, like this:

This is how we limit the background gradient to 25px in Webkit and Firefox:

1. input, textarea {
2. background: -webkit-gradient(linear, lefttop, left 25, from(#000000), to(#FFFFFF));
3. background: -moz-linear-gradient(top, #000000, #FFFFFF25px);
4. }

```
input, textarea {
    background: -webkit-gradient(linear, left top, left 25, from(#000000), to(#FFFFFF));
    background: -moz-linear-gradient(top, #000000, #FFFFFF 25px);
    }
```

For Webkit, instead of setting the end point to "left bottom," we set it to "left 25″, indicating it will end 25 pixels from the top.

For Gecko, we do the same thing by simply adding a "25px" value to the end color.

And the result is:

The second thing we want to do is create a thin white line at the top of the gradient, to give the subtle visual impression that the field is raised. How important can a single pixel be? Take a look at this article: Adding Depth with Pixel Perfect Line Work.

To create this, we'll need three points in the gradient. In the previous example, our gradient had two points: top and bottom (black→white). Here, we'll add an additional point in between them (white→black→white).

To illustrate:

How do we do this?

1. input, textarea {
2. background: -webkit-gradient(linear, lefttop, left 25, from(#FFFFFF), color-stop(4%, #000000), to(#FFFFFF));
3. background: -moz-linear-gradient(top, #FFFFFF, #0000001px, #FFFFFF25px);
4. }

```
input, textarea {
    background: -webkit-gradient(linear, left top, left 25, from(#FFFFFF), color-stop(4%, #000
    background: -moz-linear-gradient(top, #FFFFFF, #000000 1px, #FFFFFF 25px);
    }
```

In Webkit we use the **color-stop** function, but unfortunately it doesn't support values in pixels, only percentage. But thanks to paying attention to math in school we figure that 4% of 25px is 1px.

For Gecko, we simply add a third color between the first two and give it a "1px" value, indicating that it should end 1 pixel from the top.

The thin white line:

Now, let's change the black color (#000000) to a more fitting light grey (#EEEEEE):

Just some small detail work remains.

First, we'll create a darker shadow for the fields when the user hovers or selects it:

1. input:hover, textarea:hover,
2. input:focus, textarea:focus {
3.   -webkit-box-shadow: rgba(0, 0, 0, 0.15) 0px0px8px;
4.   }

```
input:hover, textarea:hover,
input:focus, textarea:focus {
    -webkit-box-shadow: rgba(0, 0, 0, 0.15) 0px 0px 8px;
    }
```

It's just an increase from 10% to 15%, but what we are after is, once again, subtlety.

The last thing we do is create some rounded corners for the button3 to further make it stand out from the other elements:

1. .submit input {
2.   -webkit-border-radius: 5px;
3.   -moz-border-radius: 5px;
4.   }

```
.submit input {
    -webkit-border-radius: 5px;
    -moz-border-radius: 5px;
    }
```

The value is the radius the corners will be rounded by. The standard border-radius is intentionally left out since Opera seems to have some problem with it.

Result:

Step 5: The Other Browsers

Now we just need to take care of the browsers that don't support CSS3 yet (IE), or only partly does (Opera).

We want the different versions (CSS3 and the normal) to look as similar as possible, and the simplest thing is to go back to the old way: images.

Simply take a screenshot of the beautiful CSS3 form and save a small portion of the gradient as an image.

Next, use it in the input and textarea as a background. As long as the CSS3 gradients comes after the background image, browsers that support CSS3 will ignore the image.

```
input, textarea {
    background: #FFFFFFurl('bg_form.png') lefttoprepeat-x;
    }
```

```
input, textarea {
    background: #FFFFFF url('bg_form.png') left top repeat-x;
    }
```

And now we are done! Enjoy your form and I hope you have learned something.

Final Preview

Chrome (4.0), Firefox (3.6), Safari (4.0):

Opera (10.50b):

Internet Explorer (8):

Full CSS

```
input, textarea {
    padding: 9px;
    border: solid1px#E5E5E5;
    outline: 0;
    font: normal13px/100% Verdana, Tahoma, sans-serif;
    width: 200px;
    background: #FFFFFFurl('bg_form.png') lefttoprepeat-x;
    background: -webkit-gradient(linear, lefttop, left 25, from(#FFFFFF), color-stop(4%, #EEEEEE), to(#FFFFFF));
    background: -moz-linear-gradient(top, #FFFFFF, #EEEEEE1px, #FFFFFF25px);
    box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    -moz-box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    -webkit-box-shadow: rgba(0,0,0, 0.1) 0px0px8px;
    }

textarea {
    width: 400px;
    max-width: 400px;
    height: 150px;
    line-height: 150%;
    }

input:hover, textarea:hover,
input:focus, textarea:focus {
    border-color: #C9C9C9;
    -webkit-box-shadow: rgba(0, 0, 0, 0.15) 0px0px8px;
    }

.form label {
    margin-left: 10px;
    color: #999999;
    }

.submit input {
    width: auto;
    padding: 9px15px;
    background: #617798;
    border: 0;
    font-size: 14px;
    color: #FFFFFF;
    -moz-border-radius: 5px;
    -webkit-border-radius: 5px;
    }
```

```
input, textarea {
    padding: 9px;
    border: solid 1px #E5E5E5;
    outline: 0;
    font: normal 13px/100% Verdana, Tahoma, sans-serif;
    width: 200px;
    background: #FFFFFF url('bg_form.png') left top repeat-x;
    background: -webkit-gradient(linear, left top, left 25, from(#FFFFFF), color-stop(4%, #EEE
    background: -moz-linear-gradient(top, #FFFFFF, #EEEEEE 1px, #FFFFFF 25px);
    box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    -moz-box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    -webkit-box-shadow: rgba(0,0,0, 0.1) 0px 0px 8px;
    }

textarea {
    width: 400px;
    max-width: 400px;
    height: 150px;
    line-height: 150%;
    }

input:hover, textarea:hover,
input:focus, textarea:focus {
    border-color: #C9C9C9;
    -webkit-box-shadow: rgba(0, 0, 0, 0.15) 0px 0px 8px;
    }

.form label {
    margin-left: 10px;
    color: #999999;
    }

.submit input {
    width: auto;
    padding: 9px 15px;
    background: #617798;
    border: 0;
    font-size: 14px;
    color: #FFFFFF;
    -moz-border-radius: 5px;
    -webkit-border-radius: 5px;
    }
```

Conclusion

That's all there is to it! With minimal effort, and the power of CSS 3, we've turned a bland and ordinary form into something beautiful. Thanks so much for reading, and feel free to ask any questions that you might have below.

Write a Plus Tutorial

Did you know that you can earn up to $600 for writing a PLUS tutorial and/or screencast for us? We're looking for in depth and well-written tutorials on HTML, CSS, PHP, and JavaScript. If you're of the ability, please contact Jeffrey at nettuts@tutsplus.com.

Please note that actual compensation will be dependent upon the quality of the final tutorial and screencast.