OmniUpdate®
Empower Web Excellence

CSS3

Cascading Style Sheets (CSS) Level 3, otherwise known as CSS3, is the latest specification available to web developers, providing new and improved tools for web design. CSS3 builds on the W3C recommended CSS 2.1.

Over the past decade, websites have transitioned from a text-oriented layout to a modular, more graphical experience. Drop shadows, gradients, background images, layout flexibility, and more were expected as part of this new experience. And with these expectations came the realization that CSS 2.1 was not built to handle expanding needs.

This primer shares how the CSS3 features detailed below deliver powerful new tools for site redesign and alterations, and improve the user experience.

Drop Shadows and Rounded Corners

Drop shadows and rounded corners have been around for quite some time; however, their execution is remarkably simple with CSS3. Prior to CSS3, drop shadows and rounded corners were handled with images, which required additional HTTP calls when loading a page. In addition, there was a lack of flexibility for developers when making adjustments to the dimensions of the container. However, CSS3 standardizes the design process with commands like `border-radius`, `box-shadow`, and `text-shadow`. Not only does this improve page load performance with the elimination of images, it increases efficient code production for developers when future adjustments are made.

Before CSS3

```
.rounded {
   background-image: url(circle.png);
   position: absolute;
   width: 25px;
   height: 25px;
}
.top-left {
   background-position: top left;
   top: 0;
   left: 0;
}
.top-right {
   background-position: top right;
   top: 0;
   right: 0;
}
.bottom-left {
   background-position: bottom left;
   left: 0;
   bottom: 0;
}
.bottom-right {
   background-position: bottom right;
   right: 0;
   bottom: 0;
}
```

CSS3

```
.example1{
   border-radius: 5px;
   border: 1px solid #E00;
}
```

Additional code: http://jsfiddle.net/JNABN/, http://jsfiddle.net/Lqxcq/, and http://jsfiddle.net/2HA9b/

Structural Pseudo-Classes

For designs that target elements in recurring locations or employ alternating styles (e.g., styling odd rows of a table), CSS 2.1 provides few practical solutions. Developers can manually style each element; however, there's a high level of maintenance to adjust the styling every time an additional item is added to a list. For a more dynamic approach, developers can use JavaScript, but a reliance on third-party frameworks like jQuery to handle dynamic styling can affect page load performance.

Before CSS3

CSS:

```
.even {
   background-color: #f3f3f3;
}
```

Javascript:

```
$("tr:even").addClass("even");
```

CSS3

```
.first-example li:nth-child(even){
   color: #f3f3f3;
}
```

Additional code: http://jsfiddle.net/5FD3P/

CSS3 introduces structural pseudo-classes, a new type of selector that increases the ability to target elements. The pseudo-class `:nth-child` provides a way to find a specific or repeating pattern of elements in a list or series. If the goal is to style every third, fourth, eighth, or nth element of a list, `:nth-child` handles it through the formula `an+b`, where `a` and `b` are integers.

CSS3

```
.second-example tr:nth-child(2n+4){
   background-color:#f3f3f3;
}
```

Additional code: http://jsfiddle.net/5FD3P/

This CSS-Tricks article provides some useful examples on how `:nth-child` can be used.

Layout Flexibility

CSS 2.1 supports the use of a stylesheet for different media types such as `print` and `screen`. However, as users increasingly access websites on mobile and tablet devices, CSS 2.1 is deficient, as it was not built to handle multiple stylesheets for the `screen` media type. CSS3 resolves this issue by introducing media queries, providing developers a way to display web content for different resolutions.

With the addition of media queries, one or more expressions can be queried to tailor a website for the current screen resolution. For example, developers can control the layout by setting a minimum or maximum width to determine styling, providing the necessary elements to achieve a responsive design.

CSS3

```
@media screen and (max-width:760px){
  body{ background-color: #333;
    }
  h1{
    color:red;
  }
  p{
    color: white;
  }
}
```

Additional code: http://jsfiddle.net/DrSRT/

Additional information on media queries and their role in responsive design is available in our Responsive Web Design Primer.

The meta tag `viewport` is also relevant when discussing media queries and responsive design. The viewport is the virtual window that displays the content of a web page currently in view, which differs between desktops, tablets, and mobile devices. With viewport, developers can configure and control the width, height, and scaling on different devices.

Browser Support

One of the major changes CSS3 introduces is the modularization of all features for W3C recommendation. Previously, if one feature was still under development while the remaining features were finished, the entire specification would be delayed for submission. Modules improve the process of recommending and standardizing individual CSS3 features as they become available. Currently, four modules have been published with formal recommendation, and more than 50 are either a candidate recommendation or a working draft.

With many CSS3 features remaining a "work in progress," how do browsers handle support for these features? And how can developers ensure design consistency for these features across multiple browsers?

Vendor-specific prefixes provide a solution for browsers to support select CSS3 features as they develop towards a stable, W3C recommended state. They allow developers to handle CSS3 features on a browser-by-browser basis. It's important to note that developers should include all available prefixes to ensure that web visitors are not limited to a single browser for optimal viewing purposes. Browsers will automatically switch to the un-prefixed element once implemented, and developers can easily adjust code by removing vendor-specific prefixes once a feature is standardized for all targeted browsers.

Popular Prefixes

-ms-	Microsoft
-moz-	Mozilla
-o-	Opera Software
-webkit-	Apple, Google

What about older browsers? How can they support any of these newer features? There are many resources available for determining browser support of newer CSS3 features. The site Can I Use describes the different levels of browser support for each feature, as well as future browser release support. Additionally, polyfills, which are considered the "spackle of the web," fill in some of the CSS3 functionality gaps for older browsers to handle rounded corners, gradients, and more. Resources such as CSS3Pie and Webshims Lib use polyfills for major CSS3 design features that are not supported by older browsers.

Additional Resources and References

1. Practical CSS3 (book by Chris Mills)
2. Responsive Web Design (book by Ethan Marcotte)
3. Prefix or Posthack (article by Eric Meyer from July 2010)
4. CSS3 (current specifications by W3C)
5. What is a Polyfill? (post by Remy Sharp from October 2010)
6. CSS3 vs. CSS: A Speed Benchmark (post by Trent Walton from April 2011)