Lesson 3 - Notes

Nathan Stilwell Feb 25, 2013

Review

Breif review of Quiz from last lesson

Images

Images are rendered in html using the `img` tag.

```
<img src="" alt="" title="" height="" width="">
```

Source

The `src` attribute is the location of the image. This URL can be a relative or an absolute path.

```
<img src="[source]">
```

Alternative

The `alt` attribute is a text description of the image. this text will be used in the case the image resource fails to load or by screen readers for accessibility.

```
<img alt="[Alternative Text]">
```

Title

The `title` attribute is used as a text description of the image, but usually shows up as a tooltip on hover of an image.

```
<img title="Title or Description">
```

Height and Width

Height and Width define the height and width of the image. While the height and width can be defined using CSS styles, providing height and width allows the browser to calculate height and width (and create the layout accordingly) before the CSS loads. Height and Width can be defined in px or percentage.

```
<img height="Image Height" width="Image Width">
```

Color

Color can be defined in the following ways in CSS

Hex Number

Hexadecimal is a base 16 numeral system (as opposed to decimal or base 10 that you are more familiar with) where number range from 0 - F instead of 0 - 9.

In decimal you count from 1 to 10 like this:

```
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
```

In hexidecimal you count from 1 to 16 like this:

```
0, 1, 2, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F, 10
```

Color in CSS is specified as a set of three numbers.

So when defining a color in hexidecimal we provide 3 numbers ranging in valued from 0 to 255. 0 in hexidecimal is 0 (or 00). 255 in hexidecimal is FF. The first number is the value of red, the second number is the value of green, and the third value is the value of blue. We designate that we are using hex to provide the color value by using a **#** symbol in front of the number set.

#ffffff

Here are a few more examples of hex colors.

`#ffffff` `#000000` `#9a9a9a` `#f0f455b` `#ff4b00` `#a30092` `#f52454`

RGB

Color is defined in RGB by providing 3 numbers in decimal to represent values for red, green, and blue.

rgb(255, 255, 255)

Here are some example colors in rgb:

`rgb(255, 255, 255)` `rgb(0, 0, 0)` `rgb(0, 244, 91)` `rgb(255, 139, 0)` `rgb(163, 0, 146)`

HSL

Using the hsl definition hue, saturation, and lightness are specified instead of a red, green, and blue values.

hsl([hue], [saturation], [lightness])

Hue is specified as a number between 0 and 360.
Saturation is specified as a percetage between 0% and 100%.
Lightness is specified as a percentage between 0% and 100%.

Here are some example colors in hsl:

`hsl(0, 0%, 100%)` `hsl(0, 0%, 0%)` `hsl(142, 100%, 48%)` `hsl(33, 100%, 50%)` `hsl(298, 100%, 32%)`

How to apply color

Color can be applied to html elements using the following CSS properties

background-color - fill color of an element
color - color of content (mostly text)

Example

```
div {
    background-color: orange;
    color: red;
}
```

div

Transparency

Opacity

The `opacity` css property changes the transparency of the entire element including content and background. `opacity` is specified as a number between 0 and 1. The number represents percent of transparency. So 1 would be 100% opaque. .5 would be 50% opaque. Here is an example:

```
.transparent-thing {
    opacity: .45; /* 45% opaque */
}
```

Alpha

Alpha compositing is the process of combining an image with a background to create the appearance of partial or full transparency. We can specify alpha in color definitions in CSS using either rgba or hsla. These definition look exactly like their non-transparent counter parts except for a fourth parameter that is a number between 0 and 1.

It looks like this.

```
/* red, 50% opaque*/
rgba(255, 0, 0, .5);

/* blue, 75% opaque */
hsla(240, 100%, 50%, .75);
```

Here's what it looks like in action:

```
.transparency-thing {
    background-color: rgba(255, 0, 0, .45);
}
```

Type

There are two general categories of css properties that apply to how type is displayed in the browser. **Font** and **Text**

font-family

To change the actual set of glyphs used in fonts, we use the font-family css property.

```
.fancy-font {
    font-family: Georgia;
}
```

`font-family` can accept a comma seperated list of fonts to try. In case a font is not available on a given platform.

```
.fancy-font {
    font-family: "Super Fancy Font", Georgia, serif;
}
```

This would try to load a font called "Super Fancy Font", if the system didn't have it, then it would try to load a font called Georgia. If it didn't have that, then the browser would make a best guess using the category "serif".

Your font-family definitions should usually end with a general fallback of

- serif
- sans-serif
- monospace
- cursive
- fantasy

font-size

The font size property specifies the size of the font. The value can either be in px, em, or %.

An `em` comes from the type settings concept of the width of a lowercase 'm' in a given font. In CSS it represents a percentage of the inherited font-size.

```
/* base font size is 16px */
.teet {
    font-size: 1em;   /* 16px */
    font-size: 2em;   /* 32px */
    font-size: 1.5em; /* 24px */
    font-size: .75em;  /* 12px */
}
```

Percentage (%) is much like `em` in that it is a size relative to it's inherited font size.

```
/* base font size is 16px */
.teet {
    font-size: 100%;  /* 16px */
    font-size: 62.5%; /* 10px */
    font-size: 150%;  /* 24px */
    font-size: 75%;   /* 12px */
}
```

font-style

Available values are *italic* and *normal*.

font-weight

Available values are **bold** and normal.

font-variant

Available values are `small-caps` and none;

text-align

The `text-align` CSS property describes how inline content like text is aligned in its parent block element. *Stay tuned to find out more about inline and block elements*

Available values are *left*, *center*, *right*, and *justified*.

text-decoration

Puts lines relative to the text.

Available values are *underline*, *overline*, *line-through*, and *none*.

text-transform

Transforms the case of the letters in an element.

Available values are *uppercase*, *lowercase*, *capitalize*, and *none*.

text-shadow

Adds a shadow to text

```
text-shadow: [x-offset] [y-offset] [blur radius] [color]
```

line-height

The height of a line of text. Can be specified in px, em, or percentage

letter-spacing

The space between letters. Can be specified in px or em.