CS105
Patrick Young
Stanford University

Appendix B: Style Sheet Reference

This section summarizes the properties that may be used within a Style Sheet. *When using Style Sheet properties, keep in mind that some web browsers may not support properties listed or values listed—this may particularly be a problem with older web browsers.*

Property Index by Section

Style sheet properties are listed in one of six sections as shown below:

Font Properties
font-family
font-style
font-variant
font-weight
font-size
font

Color and Background Properties
color
background-color
background-image
background-repeat
background-attachment
background-position
background

Text Properties
word-spacing
letter-spacing
text-decoration
vertical-align
text-transform
text-align
text-indent
line-height

Box Properties
margin-top
margin-right
margin-bottom
margin-left
margin

Box Properties (continued)
padding-top
padding-right
padding-bottom
padding-left
padding
border-top-width
border-right-width
border-bottom-width
border-left-width
border-width
border-color
border-style
border-top
border-right
border-bottom
border-left
border
width
height
float
clear

Classification Properties
display
white-space
list-style-type
list-style-image
list-style-position
list-style

Descriptions

Each property description begins with a list of one or more sets of property values. Many of the properties have keyword values—these properties must be set to specific keywords such as `bold` or `italic`. Other properties may be set to a length measurement or to a percentage—for example, `line-spacing` may be set to the number of inches, millimeters, or pixels desired between each line. Some properties may be set to a URL or a color.

In the property description, we will list keyword values in bold and value types (such as length, percentage, or color) in italics. For example, the `font-size` defines how large a font the browser will use. Possible values of `font-size` are listed as:

values: **xx-small, x-small, small, medium, large, x-large, xx-large**
larger, smaller
length
percentage

As we can see `xx-small`, `x-small`, `medium`, etc. are all listed above in bold. This means they are keyword values. We can therefore set our `font-size` to any of these specific keywords—for example:

```
li {font-size: x-large}
```

This will make all our links (created using the `<li>` tag) appear with extra-large size text.

The `larger` and `smaller` are also listed in bold. These are therefore also keyword values. They are listed on a separate line as they work a bit differently than the `xx-small`, `x-small`, `small`, etc. values. The `larger` and `smaller` keywords are relative sizes and their effect depends on the size of the surrounding text.

Both length and percentage are listed in italics rather than in bold. This signifies that they are not keywords, but rather represent a type of value that may be used with the `font-size` property. There are a number of value types commonly used by style sheet properties. These value types are described in the next section. In addition, some style sheet properties have their own specialized value types. These property-specific value types will be discussed in the descriptive text for the corresponding property.

Standard Value Types

The length, percentage, color, and URL value types are used by a variety of different style sheet properties. In this section, we describe the correct syntax to use for these value types. In addition, some style sheet properties allow a combination of values. We conclude this section with a discussion of how these combined values work.

Length

The length value type is used whenever we need to specify a measurement—for example, when defining the amount of space to place between lines or when defining margins. Style sheets allow webpage writers to specify measurements in inches, millimeters, centimeters, points, picas, and pixels (a point is 1/72nd of an inch, and a pica is 12 points). List these measurements using their abbreviations as shown in the table below:

Measurement	Abbreviation	Example
inches	in	margin-left: 2in
centimeters	cm	line-height: 1cm
millimeters	mm	word-spacing: 3mm
points	pt	font-size: 12pt
picas	pc	font-size: 1pc
pixels	px	border-top-width: 5px

In addition, length measurements can be given relative to the size of the current font. These measurements can be given using two different units: `em` and `ex`. While an `em` is traditionally the width of a capital 'M', in style sheets an `em` is the current font-size (e.g., 12 point). An `ex` is the height of a lower case 'x' character in the current font.

Percentage

Some properties may be set to a percentage. For example, the `line-height` property can be given as a percentage of the current font size.

```
.double-space {line-height: 200%}
```

Check the documentation for the specific property to determine which measurement the property is a percentage of.

Color

Style sheet colors can be defined in five different ways.

First, you may use the standard 16 HTML colors—`aqua`, `black`, `blue`, `fuchsia`, `gray`, `green`, `lime`, `maroon`, `navy`, `olive`, `purple`, `red`, `silver`, `teal`, `white`, and `yellow`. For example:

```
h1 {color: red}
```

You may also define a style sheet color using the six hexadecimal digit notation we learned in Chapter 6. For example:

```
h1 {color: #EE82EE}
```

Style sheets also support a three hexadecimal digit notation. This works exactly the same as the six hexadecimal digit notation, except the red, green, and blue components must each be between `0` and `F` instead of `00` and `FF`. A `0` means, none of that particular color is present, while an `F` represents the maximum intensity of the color.

```
p {color: #F0F}
```

This sets our `<p>` color to maximum intensity red (the first '`F`'), no green (the '`0`'), and maximum intensity blue (the third '`F`'). Mixing red and blue gives us purple.

You may use a special RGB notation and specify the actual decimal numbers between 0 and 255 for red, green, and blue. This works exactly the same as the standard HTML six-hexadecimal digit notation, except no hexadecimal conversion is required. For example:

```
p {color: rgb(255,0,255)}
```

This is the equivalent to our previous <p> style definition.

Finally, you may use the same RGB notation, except providing percentages:

```
p {color: rgb(100%,0%,100%)}
```

URL

In some cases, you may provide a URL as the value of a property—to provide the location of an image file for use as a background, for instance. Here is an example of the URL property:

```
p {background-image:
        url(http://www.statecollege.edu/images/logo.gif)}
```

URLs may be absolute or relative. Relative URLs are relative to the location of the style sheet. If your style sheet is included directly in your HTML file using a <style> tag, this won't be an issue. However, if you include a style sheet using the <link> tag, keep in mind that all URLs in the sheet will be relative to the *.css file *not* relative to the *.html file.

Combinations

Some properties have values which are actually combinations of other property values. The font property, for example, allows the user to set font-style, font-variant, font-weight, font-size, line-height, and font-family all on one line.

```
p {font: italic 10pt/12pt sans-serif}
```

sets the font for each paragraph to italic, sans-serif 10-point font with a 12-point line height. The border-width property allows the user to set border-top-width, border-right-width, border-bottom-width, and border-left-width all at once.

```
table {border: thick thin thin thin}
```

sets the border around each table with a thick top border, and thin right, bottom, and left borders.

Properties which are combinations of other properties will be marked as *combination*. The rules for combining properties will be included in the property's descriptive text.

Font Properties

font-family

values*:* *fontname*
serif, sans-serif, cursive, fantasy, monospace
list-of-fonts

The font-family property is used to set the type of font used. It may be set to a specific font name such as Arial, Helvetica, or Times or to one of five generic font names—

serif, sans-serif, cursive, fantasy, and monospace. Font names containing spaces must be enclosed in quotes—for example "Times New Roman".

Instead of listing a specific font name or generic font name, the font-family property may also be set to a list of font names and generic font names. Font names in the list should be separated by commas. The browser will use the first font on the list which is available on the computer. For example:

```
body {font-family: "New Century Schoolbook", Times,
                     "Times New Roman", serif}
```

will try to set the font to New Century Schoolbook. If that font isn't found, it will try Times, followed by Times New Roman. If none of these fonts are found, it will use any font with serifs which is available.

font-style

values: **normal, italic, oblique**

Font style may be set to normal, italic, or oblique. If the font-style is set to italic, the browser will attempt to display the text in italics. Similarly, if the font-style is set to oblique, the browser will attempt to display the text as oblique. Oblique fonts are slanted and are similar to italic fonts.

font-variant

values: **normal, small-caps**

Font variant may be set to normal or small-caps. On web browsers supporting the font-variant property, using small-caps will transform text from upper- and lower-case letters to small capital letters.

font-weight

values: **normal, bold**
bolder, lighter
100, 200, 300, 400, 500, 600, 700, 800, 900

The font-weight property can be set to several different types of values. First, font-weight can be set to either of the keywords bold or normal. The font-weight can also be set to bolder or lighter, these settings will increase or decrease the weight of the current font relative to the surrounding text. Finally, the font-weight can be set to one of nine different numeric weight settings—100, 200, 300, 400, 500, 600, 700, 800, or 900. The 100 setting is the lightest available setting, and the 900 is the heaviest. 400 corresponds to normal weight text and 700 corresponds to the standard bold weight.

font-size

values: **xx-small, x-small , small, medium, large, x-large, xx-large**
larger, smaller
length
percentage

The font-size property, naturally determines the size of text. It can be set to one of several types of values.

It can be set to one of seven set sizes: xx-small, x-small, small, medium, large, x-large, xx-large. Like the standard HTML sizes 1-7, the actual font size used for these sizes will depend on the user's default font size preferences.

The font-size property may also be set to the relative sizes—larger and smaller. These sizes are relative to the font-size of the enclosing HTML element. For example, if a <td> with style="font-size: larger" is placed within a <table> with style="font-size: small", our <td> will display font of medium size while the other table elements will use small font.

Finally font-size may be set to a specific length, such as 12pt, or to a percentage of the size used by enclosing HTML elements. Length and percentage follow the standard style sheet syntax discussed at the beginning of this appendix.

font

values: *combination*

The font property can set font-style, font-variant, font-weight, font-size, line-height, and font-family all at once. In its simplest form, the font property lists the value for font-size followed by the value of font-family (in that order). For example:

```
h1 {font: xx-large Helvetica}
h2 {font: 14pt "New Century Schoolbook",
                          "Times New Roman", serif}
```

We can add a line-height value after the font-size, with a slash '/' separating the two (the line-height value is discussed in the section on text properties below). Here is an example which sets both line-height and font-size:

```
p {font: 12pt/14pt Helvetica}
```

This sets all paragraphs to display 12 point Helvetica with 14 point line spacing.

Finally, we can precede the font-size with values for font-style, font-variant, and font-weight. These three items may be listed in any order (although they must come before font-size) and they are all optional. For example:

```
p  {font: italic bold 12pt/14pt Helvetica}
li {font: bolder small "Times New Roman", serif}
h3 {font: 700 120% Helvetica, Arial, sans-serif}
```

Color and Background

color

values: *color*

The color property determines the text color. This property may be set to any of the color values discussed in the Standard Value Types section at the beginning of this appendix. Link text colors are handled using the :link, :visited, :active, :focus, and :hover pseudo-classes described in Chapter 6.

background-color

values: *color*
transparent

The background-color property may be set to a color value or to the keyword value transparent. If it is set to transparent, whatever color or background is behind the element will be displayed. The transparent setting is the default behavior.

background-image

values: *url*
none

The background-image property may be used to display an image behind the element. This property is set to the URL of the image to display, for example:

```
table {background-image: url(stanford.jpg)}
```

See the Standard Value Types section at the beginning of this appendix for more information on legal URL values. The background-image property may also be set to the keyword value none—this is the default behavior.

background-repeat

values: **repeat, repeat-x, repeat-y, no-repeat**

The background-repeat property determines what happens if the image set with the background-image property isn't big enough to fill the entire webpage. The default behavior, repeat, tiles the image horizontally and vertically—creating as many duplicate images as needed to fill the webpage. The repeat-x setting tells the browser to tile the image horizontally, but not vertically. Similarly repeat-y tells it to tile vertically, but not horizontally. The no-repeat tells it not to repeat at all.

background-attachment

values: **scroll, fixed**

The background-attachment property determines what happens when the user scrolls the webpage. If it is set to scroll, the background image scrolls along with the webpage. This is the default behavior. If background-attachment is set to fixed, the background will remain fixed and the webpage will appear to scroll in front of the image.

background-position

values: *length-pair*
keyword-pair
percentage-pair

The background-position property can be used to place the background image relative to the enclosing element. It can be given as a pair of length measurement, as a pair of keywords, or as a pair of percentages. The first item in the pair determines horizontal placement of the background image and the second item determines vertical placement. Items in the pair are separated by spaces.

Length measurements provide the distance from the top-left corner of the element to the top-left corner of the image. For example, the following style rules place the top-left corner of the "logo.gif" image one inch from the left-side of the webpage and two inches from the top of the webpage:

```
body {background-image: url(logo.gif);
      background-position: 1in 2in}
```

Notice that there is a space, not a comma, between the 1in and 2in measurements.

Keyword pairs use the keywords top, center, and bottom for vertical placement and left, center, and right for horizontal placement, for example:

```
background-position: right bottom
```

Percentages place the image relative to the width and height of the enclosing body. The actual percentage given is the percentage of width or height where the enclosing body and image will match. For example:

```
background-position: 50% 100%
```

tells the browser that the 50% point on the image horizontally should match the 50% point of the enclosing element. This will center the element horizontally. The 100% tells the browser that the 100% point on the image vertically (that is to say the bottom of the image) should match the 100% point on the surrounding element (that is to say the bottom of the element). This aligns the element to the bottom of the surrounding element.

You may provide a mixed pair of length, keyword, and percentage values. For example:

```
background-position: 50% 2in
```

You may also provide only a single value. In this case, the browser will use the value for horizontal placement and will assume a center vertical placement.

background

values: *combination*

The background property can be set to any combination of background-color, background-image, background-repeat, background-attachment, and background-position. These items may be listed in any order. Here is an example:

```
background: url(logo.gif) no-repeat center center fixed
```

Text Properties

word-spacing

values: *length*
normal

As its name implies, `word-spacing` controls the space between words. The value given is the amount the word spacing is increased from the normal browser word spacing. The value given may be negative.

letter-spacing

values: *length*
normal

The `letter-spacing` property controls the spacing between letters within each word. The amount given is the increase desired from the normal browser letter spacing. The amount given may be negative.

text-decoration

values: **none, underline, overline, line-through, blink**

Text decoration may be used to add lines above, below, or through the text. Text decoration may also be used to blink the text on and off. Multiple values may be listed simultaneously. Here is an example using the `:hover` pseudo-class (described on pages 6-49 to 6-50):

```
:hover {text-decoration: underline overline}
```

This will create lines above and below a link when the mouse is moved on top of them.

vertical-align

values: **baseline, sub, super, text-top, middle, text-bottom**
top, bottom
percentage
length

This property controls vertical alignment of an element.

The `baseline`, `sub`, `super`, `text-top`, `middle`, and `text-bottom` settings control alignment relative to the parent element. They may be used to align the top of the element to the top of nearby text (using `text-top`), to `middle` align text, or to align the bottom of the element with the bottom of nearby text (using `text-bottom`). The `baseline` setting aligns text to the baseline—the baseline is similar to the bottom of text, except it does not take descending letters into account.

The `sub` and `super` values turn the text into a subscript or superscript.

The `top` and `bottom` settings control alignment within the element itself. Setting `vertical-align` to `top` will align the top of all items in the element to the top of the tallest element. Similarly setting it to `bottom` will align the bottom of all items to the bottom of the tallest element.

The percentage and length settings raise or lower the element with respect to the surrounding elements. The percentage setting is relative to the current line height.

text-transform

values: **`capitalize`**, **`uppercase`**, **`lowercase`**, **`none`**

The `text-transform` property may be used to capitalize all words within the selected element, or to turn all words to all uppercase or all lowercase.

text-align

values: **`left`**, **`right`**, **`center`**, **`justify`**

The `text-align` attribute controls horizontal alignment of text.

text-indent

values: *`length`*
`percentage`

This property controls indentation of text. It may be given as a length or a percentage. Percentages are relative to the width of the parent element, for example, the width of a table cell or the width of the entire window.

line-height

values: **`normal`**
`length`
`multiplier`
`percentage`

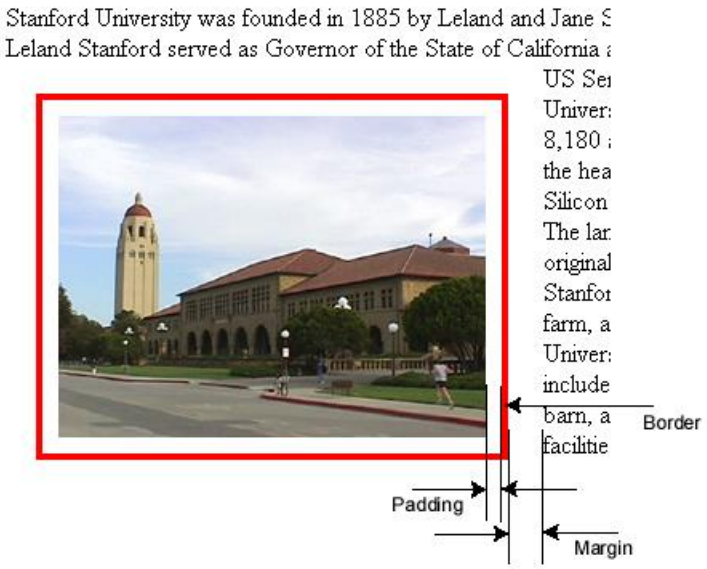
The `line-height` property controls spacing between lines of text. The `line-height` may be given as a length measurement, as a multiplier number, or as a percentage of the font size of the element. Multiplier numbers give multiples of the normal line height. For example,

```
line-height: 3
```

would signify three times the current line height.

Box Properties

Style sheets allow us to specify margin, padding, and border around elements. The padding is the distance between an element and its border. The margin is the distance between the border and surrounding text.



margin-top, margin-right, margin-bottom, margin-left

values: *length*
percentage
auto

These four properties set the margin widths for an element. They may be set to a specific length or as a percentage of the enclosing element.

margin

values: *combination*

The `margin` property allows us to set all four margins using one line. If only one value is specified, then that value will be used for all four margins. If two values are specified, then the first value is used for the top and bottom margins and the second value is used for the right and left margins. If all four values are specified, they should appear in the order: top, right, bottom, and left. Here are some examples:

```
margin: 5px             /* all margins are 5 pixels */
margin: 0.2in 0.5in     /* top and bottom margins are 0.2",
                           right and left margins are 0.5" */
margin: 5px 2px 1px 3px /* top margin is 5 pixels
                           right margin is 2 pixels
                           bottom margin is 1 pixel
                           left margin is 3 pixels */
```

padding-top, padding-right, padding-bottom, padding-left

values: *length*

percentage

These four properties set the padding widths for an element. They may be set to a specific length or as a percentage of the enclosing element. However, see `padding` warning below.

padding

values: *combination*

The `padding` property allows us to set all four padding values at the same time. It uses the same format as the margin property described above.

Warning: The `padding-top`, `padding-right`, `padding-bottom`, `padding-left`, and `padding` properties are supported on only a few elements in Internet Explorer. Supported elements include the `<span>` and `<div>` tags and the table elements. Padding is not supported on the `<img>` tag.

border-top-width, border-right-width, border-bottom-width, border-left-width

values: *length*
thin, medium, thick

These four properties determine the width of the border. They may be set to specific lengths or to one of three keywords—`thin`, `medium`, or `thick`.

border-width

values: *combination*

The `border-width` property allows us to set all four border width values at the same time. It uses the same format as the `margin` and `padding` properties described above.

border-color

values: *color*
color-combination

The `border-color` property sets the color of the border. Each border (top, right, bottom, and left) may have its border color set independently. Specifying a single color sets the border color for all four borders. Multiple color combinations follow the same rules as the `margin` and `padding` properties. Specifying two colors sets the border for the top and bottom border and the right and left border. Specifying all four colors sets the border color for top, right, bottom, and left (in that order).

Warning: Setting `border-color` by itself will have no discernable effect, unless the `border-style` propery below is also set.

border-style

values: **none, dotted, dashed, solid, double, groove, ridge, inset, outset**

This property determines the appearance of the border. The default setting is `none`. Note that not all settings are supported by all browsers.

border-top, border-right, border-bottom, border-left

values: *combination*

Each of these four properties can be used to set `border-width`, `border-color`, and `border-style` all on one line. The values may be listed in any order. Here is an example:

```
border-top: red thick solid
```

border

values: *combination*

The `border` property allows the user to set `border-width`, `border-color`, and `border-style` for all four borders all at once. This property may only be used if the width, color, or style specified is the same for all four borders. Here is an example:

```
border: thin double blue
```

This sets all four borders to thin, blue, double lines.

width

values: *length*
percentage
auto

This property controls the width of the element. It may be set to a specific length or to a percentage of the parent element. For images, setting the `height` property to a specific height and the `width` to `auto` will instruct the browser to calculate the proper width of the image to maintain the image's original aspect ratio.

height

values: *length*
auto

This property controls the height of the element. For images, setting the `width` property to a specific width and `height` to `auto` will instruct the browser to calculate the proper height of the image to maintain the image's original aspect ratio.

float

values: **left, right, none**

This property can be used to control text flow around the element. If it is set to `left`, the element will appear on the left side of the webpage, with text flowing on the right. Setting it to `right`, places the element on the right with text flow on the left.

clear

values: **none, left, right, both**

This property determines whether or not other elements are allowed to float alongside the element. If clear is set to `left`, no floating images or other floating elements will be placed to the left of the element. Instead, the element will be placed below the image. The `right` and `both` settings work similarly.

Classification Properties

display

values: **`block`, `inline`, `list-item`, `none`**

This rather unusual property can turn HTML tags which normally create blocks of text, such as the `<h1>` tag, into inline text tags—eliminating the carriage returns normally created by these block items. It also can convert inline tags, such as the `<i>`, `<b>`, or `<u>` tags into block level tags.

The `list-item` setting works exactly the same as the block setting, except that a `list-item` graphic, such as a bullet '•' will precede the item when displayed.

The `none` setting tells the browser not to display the item at all. This may be used, for example, to instruct the browser not to display images:

```
img {display: none}
```

white-space

values: **`normal`, `pre`, `nowrap`**

As we have learned, HTML normally treats all whitespace the same. If we separate the words in an HTML file by a single space or by fives spaces and a tab, the browser will display our webpage the same. Excess whitespace is ignored. However, if we set the `white-space` property to `pre`, we are telling the web browser to treat our text as preformatted, and to display whitespace as originally written—displaying all spaces, tabs, and carriage returns as indicated in the original HTML file.

The `nowrap` setting works similar to the `pre` setting, except text will not wrap when it is too long to fit on a single line. Line breaks will only occur when explicitly marked with `<br>` tags.

list-style-type

values: **`disc`, `circle`, `square`, `decimal`, `lower-roman`, `upper-roman`, `lower-alpha`, `upper-alpha`, `none`**

The `list-style-type` determines whether bullets (discs), circles, or squares are used to indicate list items. Or what kind of numbering scheme the system should use for ordered lists.

list-style-image

values: *`url`*
`none`

Instead of the standard disc, circle, or square list indicators, you may specify your own image file to use with lists. The URL supplied should be the location of an image file. For example:

```
li {list-style-image: url(tree.gif)}
```

will place the image "`tree.gif`" in front of every list item.

list-style-position

values: **`inside`**, **`outside`**

This property determines whether or not list indicators, such as discs or squares are indented.

list-style

values: *`combination`*

This combination property allows the webpage author to specify `list-style-type`, `list-style-image`, and `list-style-position` values all on one line. The property values may be listed in any order, for example:

```
list-style: square outside
```