

**Cascading Style Sheets:
A History, Review and Demonstration**

**By
Joe Panico**

j_panico@cox.net

**TS5504 Technical Writing Course
Instructor Bill Akins**

**Capella University
Technology Department**

May 20th, 2005

Abstract

The technology of the World Wide Web as invented in 1989 and burst on the world scene in 1993. The Hypertext Markup Language (HTML) used to code web pages burst with it. Because HTML is limited in its capabilities, as it was intended to be, the browser manufactures forced its subsequent development to include styling effects. Fortunately the World Wide Web Consortium, a governing forum for the Web, developed Cascading Style Sheets (CSS) language to provide style and lend creative possibilities to web design. After a war between browser manufactures, CSS is now taking its place as the new language to create professional web sites. Its use with XHTML makes CSS the cutting edge of web technology, leading the Web toward global accessibility and aesthetic beauty.

Table of Contents

Cascading Style Sheets: A History, Review and Demonstration

	Abstract	2
	Table of Contents	3
I.	History of the World Wide Web	4
	A. The Beginning of the World Wide Web	4
	B. Tim Berners-Lee Contributions	4
	C. The Internet and the Web	5
	D. The Initial Purpose of the Web	5
	E. The Main Forces Surrounding the Web	6
	1. World Wide Web Consortium	6
	2. Browser Manufacturers	6
	3. Web Developers	7
	4. The Public	7
II.	The Emergence of CSS and XHTML	8
	A. Resolution and a New Beginning for Web Development	8
III.	Advantages of CSS	8
	A. CSS Saves Time: Simplifies Editing	8
	B. CSS Eliminates the Need for Tables as a Layout Method	9
	C. More Style Features	9
	D. Accessibility	9
IV.	CSS: Understanding the Basics	9
	A. Purpose of the Review	9
	B. Basic Syntax	10
	C. External, Internal and Inline Code	10
	D. Meaning of Cascading	11
	E. Containers: Parent-Child Relationship and Inheritance	12
	F. The Property Groups	12
	G. The Units of Measurement for Value Lengths	13
	H. Browser Conformity	13
	I. Companion Web Site	13
V.	Property Groups: A Review with Examples	13
	A. Font Properties	13
	1. Shorthand Method for Writing Property Definitions	14
	B. Text Properties	14
	C. Class Selectors	15
	D. Background Properties	15
	E. The Box Property Groups – Margins, Padding and Borders	16
	F. Margin Properties	16
	1. Margin Flow	17
	G. Width and Height	17
	H. Padding Properties	17
	I. Border Properties	18
	J. Link (Anchor) Styles	18
	K. Position Properties	19
	1. Z-Index	20
	L. Layout Without Tables	20
	M. Conclusion	20
	Appendix A	21
	Appendix B	23
	References	25

Cascading Style Sheets: A History, Review and Demonstration

This paper provides, in part, an account of the evolution of web page development starting with the humble beginnings of the World Wide Web to the implementation of Cascading Style Sheets (CSS). CSS version 2 is the current culmination of web page authoring language. Its development was concurrent with Hypertext Markup Language (HTML), but has been more slowly adopted and used as a standard within the web development community. While HTML provides the structure of web pages, CSS provides the style or design elements to web pages. After the historical review, the remainder of this paper will explain and demonstrate some of the more common components the CSS language. For the latter half of this paper, it is assumed the reader has some experience in constructing web pages with HTML.

History of the World Wide Web

The Beginning of the World Wide Web

There is a common misunderstanding that the Internet was created in 1990 by an English physicist at CERN laboratories in Switzerland. Actually, there had been intercontinental Internet systems in place for over a decade before software engineer Tim Berners-Lee created his now famous contributions to Internet technology while at CERN in 1989-90. Tim Berners-Lee did create the technology for the World Wide Web (WWW, or Web), which rendered a more advanced Internet system that eventually connected the entire world of computers. What is less commonly understood is that the *idea* for a WWW existed as far back as 1945 long before its physical and software technology existed (W3C, 1998). Computers were first networked in small but separate network systems in the 1960's. In 1978 these independent computer networks were joined into the first Internet, that is, a network of computer networks, owing to the development of TCP/IP technology by Vint Cerf. (Griffin, 2000). TCP/IP is a protocol allowing communication between networks of different platforms and operating software.

Tim Berners-Lee's Contributions

Tim Berners-Lee wanted to improve the existing means of computer communication at the CERN facility. The Web technology he created consisted of five innovations that would allow document data to have continuous accessibility without its author having to send the data per each request from another user somewhere else on an Internet. In other words, documents (or web pages as we call them today) could now be stored on a server and retrieved (viewed) at any time independent of their authors' assistance. According to researcher Griffin, Berners-Lee's contributions were:

- 1) HTTP - Hypertext Transfer Protocol – a means to transfer files on an Internet.
- 2) HTML - Hypertext Markup Language - the language that constructs web pages.
- 3) URI - Today call Uniform Resource Locator (URL) – the format for web addresses.
- 4) WWW Browser - Software to display web documents.

He called his browser the World Wide Web, which of course later became the umbrella term for the entire phenomena of online Web content and communication.

- 5) Server Software - Software on the host computer to serve or send the web documents to visitors.

With this advance in technology a new Internet system was born, and with the advent of the Mosaic browser (1994) it grew into a global network of communication. Its content became accessible throughout the world and became known as the World Wide Web. Computer hobbyists could quickly champion HTML and become webmasters. Information could be accessed instantaneously across the globe. Now data and documents were readily available for viewing simply by pointing a browser to the URL of the web page. Hypertext linking became available. The computer science concept of hypertext media, the ability of a document to provide immediate accessibility of data and resources referenced in the content of the document, became a reality. The Information Age arrived.

Now the Internet supported the World Wide Web as we know it today, a dream first expressed by computer visionaries as early as 1945 (Griffin).

The Internet and The Web

Though public jargon uses the words Internet and World Wide Web interchangeably, Tim Berners-Lee makes a distinction between the two entities:

The Web is an abstract (imaginary) space of information. On the Net, you find computers -- on the Web, you find document, sounds, videos... information. On the Net, the connections are cables between computers; on the Web, connections are hypertext links. The Web exists because of programs which communicate between computers on the Net. The Web could not be without the Net. The Web made the Net useful because people are really interested in information (not to mention knowledge and wisdom!) (Berners-Lee, T. 1998).

The Initial Purpose of the WWW

Tim Berners-Lee and his associates envisioned the WWW to be based on a free and universally accessible Internet system of global magnitude, used primarily for the dignified pursuit of academic research and humanistic and cultural record. The founders did not initially imagine the Web to be used for commercial profit, controversial material or even necessarily for personal use. Regardless of its early purpose, the Web dispersed across the globe for all people and sectors of society. Its content and uses diverged in myriad directions. Though the founders did not try to control the content explosion on the Web, they did intend to control the computer language standards within the web programming and web authoring professions.

Berners-Lee and others were concerned that the underlying foundation of the Web should not fracture into vying factions, thereby, creating separation and limited accessibility of the technology. The WWW technology had to remain universal and the many forms of language code supporting it needed to be standardized. To this purpose Berners-Lee and the founding custodians of the WWW established an overseeing World Wide Web Consortium (W3C) that would research and develop the language and technology of the Web for the purpose of making recommendations to the programming and web authoring communities. The recommendations are designed to keep the Web universal in both accessibility and professional standards. The W3C's purpose includes preventing the WWW from being dominated by one or more entities. Through the W3C, the Web is becoming an organized, self-governed community (W3C, 2004).

The Main Forces Surrounding the Web

This technology was embraced with a seemingly exponential growth by millions of individuals across industries, continents and cultures. There were four forces involved in its spontaneous growth:

- 1) The small custodial community of founders and participating professionals that now comprised the W3 Consortium, whose purpose was (and still is) to keep the WWW free, cohesive and standardized.
- 2) Browser manufacturers.
- 3) Web authors and developers, many of whom came from the desktop publishing, graphic design and programming fields.
- 4) The public at large with its demands and expectations for the Web to grow in all directions of e-commerce, professional and personal uses.

World Wide Web Consortium

CERN laboratories soon learned it could not manage the WWW while conducting science, therefore, it relinquished its position to Berners-Lee and his newly founded W3C organization in 1993. The W3 Consortium held its first meeting in Paris in 1994. Soon it started a second headquarters at MIT, where Berners-Lee now lives and is director of the organization.

The W3C is comprised of any individual who wants to be involved in the process of defining the standards and direction of the WWW. Many corporations (Microsoft, IBM, AOL, for example) have representatives involved in the W3 Consortium committees participating in the design of web technology standards.

Perhaps a distinction should be made. It is interesting to note the difference between a standard and a recommendation. The W3 Consortium actually makes recommendations for web development conformity. As these recommendations are adopted and practiced by the development community they become the “standards” for the industry. The W3C has no authority. It only researches and recommends. (Holzschlag, 2002) Compliance is voluntary, but yet, is increasingly understood across professions to be the only way to safeguard an organized and universal Web. Referring to the web development community Molly Holzschlag, a noted author and developer writes “We’re trying to make sense out of the chaos of the past years and achieve some stability in our designs, tools and methods. It makes sense: we’re looking to find some conventions to make our jobs easier” (2002, p. 4).

Browser Manufactures

The browser manufacturers, of course, needed to make profit and compete for public popularity. Mosaic (later Netscape) was the first graphical, point and click, browser that instantaneously became popular in 1994. A year later Internet Explorer appeared on the scene. The browser programmers created their own proprietary HTML code that created web style effects for the purpose of winning over web authors and end users to their browser. Known as “browser wars”, the competition created factions in the development community and reduced the universality of the Web. The public had to choose browsers, and webmasters created web sites specifically for one browser or another. Browser programmers developed their code outside of the W3 Consortium protocol. They were working against standardization, by creating proprietary code all in the interest of winning market share.

Though this competition may have secured profits, it also caused frustrations and discouragement (as well as added time and expenses) among the web development and design professionals. Web authors wanted code standards that would work across all browsers: the very goal of the W3 Consortium. Eventually, browser manufacturers were lured into the W3C forums and assisted in working out their differences as they participated in creating the W3C recommendations: a transitional process that is still occurring today (Schengili-Robers, 2004).

Web Developers

The webmasters and developers wanted a standard of language code that worked seamlessly with all browsers. Moreover, they requested greater flexibility and variation in their ability to code traditional typographical and design elements into their web layouts. Many webmasters are used to creating documents with the prevailing desktop publishing and graphics software such as PageMaker, InDesign, Photoshop and Illustrator. Text and font characteristics, paragraph and margin formatting, background qualities, text boxes, images, object positioning, color, general layout and even interactive functionality (which is beyond the scope of this paper), were all design elements web authors desired to control.

To support the W3C recommendations, developers organized the Web Standards Project. Its purpose is to promote web standards (not the least of which includes CSS) among their peers and to the browser programmers specifically.

Founded in 1998, The Web Standards Project (WaSP) fights for standards that reduce the cost and complexity of development while increasing the accessibility and long-term viability of any site published on the Web. We work with browser companies, authoring toolmakers, and our peers to deliver the true power of standards to this medium (WaSP, 1998).

The Standards Project is successful. More and more, browsers are conforming to standards. Gradually the web authoring community is complying with W3C recommendations.

The Public

The public wanted the Web to do everything: ecommerce, personal home pages, business, banking, shopping, academia, government, religion, search engines, databases, societies, libraries, controversial content, news, politics, and much more. The entire fabric of society is represented on the Web. Careers, dreams and passions all revolve in and around the World Wide Web. Public demand accentuated the need for standardization. They too wanted the Web to be seamless, visual, functional and free of browser wars. However, in their frenzy to learn HTML and try new style code, the public inadvertently encouraged the browser manufacturers to continue to ignore CSS, the authentic Web styling language. In their reluctance to implement CSS, manufacturers provided new proprietary HTML code, hence fortifying the continuation of the browser wars.

The Emergence of CSS and XHTML

Resolution and a New Beginning for Web Development

While browser manufacturers were in a furry, the W3C patiently dealt with the tension. Over a six-year period the W3C made recommendations that quelled the forces considerably. It recommended the end of HTML development as such and the emergence of the style language CSS to provide versatile and powerful styling elements that HTML cannot. HTML was no longer being developed beyond version 4.01. Instead XHTML was created, which is HTML 4.01 incorporating code procedures from XML (Extensible Markup Language), a structured data language (Holzschlag, 2003).

These recommendations would solve two problems. It stopped HTML from being used as a style language. Now the standard is to not create new HTML code, but rather, use CSS as a means to format style and design for web page construction. This pleased the web developers. Now they can use CSS to add the style they wish. It sent a strong message to browser manufacturers that the days of manipulating HTML code for public popularity was over. Instead they should focus on complying with CSS (and other important recommendations). Both the XHTML and CSS recommendations further added accessibility of web content to mobile, multi-media and electronic devices such as TV, PDAs, cell phones, Braille and text readers (Holzschlag).

Interestingly, CSS was first recommended early in the WWW timeline in 1994. Hakon Wium Lie recognized the need to create a style language for web pages even before the browser was started. Whereas HTML proliferated suddenly with the advent of the Web, CSS slowly became known to the development community and to the general public. CSS was promoted quietly through the W3C. HTML was trumpeted brashly by the Web itself, overshadowing the importance (even the awareness) of CSS. Finally now that the din is over, CSS is taking its place as the standard of web page design.

A quote from Hakon Wium Lie, an early developer of CSS:

So, the motivation for developing style sheets was twofold: we wanted to give authors the presentational effects they craved, while stopping HTML from sliding down the ladder of abstraction to become a presentational language (WaSP, 1998).

Advantages of CSS

CSS Saves Time: Simplifies Editing

By using one set of style rules, that is one CSS file, an author can control and edit hundreds of pages in a large web site with just a slight edit of style rules.

CSS is a breakthrough in Web design because it allows developers to control the style and layout of multiple Web pages all at once. As a Web developer you can define a style for each HTML element and apply it to as many Web pages as you want. To make a global change, simply change the style, and all elements in the Web are updated automatically (W3 Schools, 1999).

CSS Eliminates the Need for Tables as a Layout Method

Tables, which first appeared in HTML 4, were meant to provide data cells on a web page in much the same way as an Excel sheet. However, web developers discovered that by nesting tables within tables and making complex tables with row and column manipulation, they could finally begin to place images and content on a page with more versatility and accuracy. This gave HTML a convoluted and even unintended ability to add advanced layout and element positioning to web pages. The drawback was that tables are awkward and complex to use for advanced styling purposes. Tables are difficult to edit. Each table cell needs its own formatting (Shafer, 2003).

CSS on the other hand can produce advanced page layout without tables. Webmasters are turning to CSS rules as the preferred means to design page layout. CSS simplifies the source code and the editing process. It also reduces construction and design time.

When tables are used, and styled with CSS, their editing is much easier. By simply changing the table style rules the corresponding table formatting is adjusted across the site

More Style Features

CSS as a language was designed to offer more style features to choose from with which to create design effects. Web developers are still learning new design possibilities, layout methods and styling effects through advanced CSS use. The early dream for web developers to have the similar capabilities in web design as they do with their advanced desktop publishing software applications has come about. In fact, the potential use and effects of CSS are still uncharted. (Holzschlag, 2002).

Accessibility

CSS, along with XHTML allows web page content to be accessible on multi-media devices like TV, aural and Braille reading, PDA, cell phones, and other mobile devices. CSS is essential to the fulfillment of the Section 508 Accessibility Act, which mandates publicly funded electronic media to be equally accessible for the sense-impaired (Holzschlag).

CSS: Understanding the Basics

Purpose of This Review

There are many books on CSS filled with hundreds of pages of information targeting basic to professional themes of style authoring skills. The remaining portion of this paper will explain only the basic syntax and properties of CSS. It is not intended to be an exhaustive review. It is a limited attempt to review the common style properties that are supported by **Internet Explorer 6.0**. For a complete explanation of Cascading Style Sheets, I refer the readers to the many books on the subject, written by professional web developers.

Basic Syntax

CSS and HTML are both relatively easy languages to use. Just as HTML has three elements to its syntax so does CSS: **selectors**, **properties** and **values**.

HTML tag	=	command	+	attribute	+	value		inside <>
example		font		face		"verdana"		
CSS rule	=	selector	+	property	+	value		inside { }
example		p		font-family:		verdana;		p {font-family: verdana;}

HTML tags are enclosed in brackets < >. A CSS rule consists of a selector followed by a style definition that is enclosed in curly brackets { }.

HTML tags can be defined by attributes with values. Similarly CSS rules are defined by properties with their values. Properties are defined by values. Example: the property "color" is defined by the value "blue", or the property font-family is defined by the value "arial". A property is followed by a colon (:) and a value is followed by a semicolon (;).

The values and the properties they define are together called the "definition" for the style rule (Teague, 2001). An example:

CSS rule with three properties: **p {font-family: verdana; color: blue; font-size: 14px;}**

This CSS rule uses the HTML paragraph command p as its selector and is defined by the three properties font-family, color, and font-size. The three properties are defined by the values of Verdana, blue and 14 pixels, respectively. Both the properties and values, inside the curly brackets, comprise the "**definition**" of the p style rule.

An HTML tag can be modified by only a few attributes. However, a CSS rule can contain many properties, hence providing the webmaster a greater range of styling effects and creative design elements.

External, Internal and Inline Code

CSS properties can be applied to web pages in three ways. Properties can be written **inline**, that is, inside the HTML tag brackets in the body of the source code. Or, they can be written **internally** between the opening and closing of the <head> tags. The third option is to use an **external** style sheet, which is a separate CSS document (using a .css extension) linked to web pages by a <link> tag between the opening and closing <head> tags.

It is this third option that refers to the word **Sheet** in Cascading Style Sheets. A Cascading Style Sheet is literally a CSS file listing CSS rules that has a .css file extension. These CSS files are linked to the web pages whose design elements they define (Teague).

The following is an example of adding a CSS rule **inline** within a <p> tag:

<p style="font-family: verdana; font-size: 16px; color: blue;">

Four score and seven years ago our fathers brought forth.... </p>

In the above inline CSS rule the text will display the style properties of Verdana font, at size 16 pixels and color blue.

CSS rules can be applied **internally** by writing them between the opening and closing <head> tags. The internal method requires an opening and closing <style> tag inside the head tags as shown in the following example:

```
<head>
<style type="text/css">
p {font-family: arial; font-size: 18pt; color: red;}
</style>
</head>
```

The above rule defines the <p> tag as having the properties of Arial font, at size 18 points, color red. As the <p> tag is used in the body of the source code, it will apply these style properties to text it encloses.

To provide an **external** CSS file (example: pagestyle.css) a link tag specifying the external sheet's name and directory path is also needed between the <head> tags.

```
<head>
<link rel="stylesheet" href="../style/name-of-file.css">
</head>
```

Notice the <style> tag is not needed. In the above example, the CSS file is inserted in place of the "name-of-file.css". A name of a CSS file might be "pagestyle.css", or "home.css".

By this <link> tag, the browser knows to find the .css file with the name specified in the directory path. The file itself will only have CSS rules listed in it. There is added benefit to using an external CSS file. If a web site has hundreds of web pages and all are linked to one .css style page for their style element (effects), then to change a style element throughout the web site only requires changing the style rule for that element in the external .css file. This is quite advantageous for busy webmasters.

Meaning of Cascading

A web site can use all three means of applying style rules, with inline rules having priority over internal rules, and internal rules having priority over external sheet rules. External style rules will apply to a web page if there are no subsequent internal or inline rules overriding them. Hence, there is a "**Cascading**" effect of style application from external to inline levels. As the w3schools.org (1999) site explains:

Generally speaking we can say that all the styles will "cascade" into a new "virtual" Style Sheet by the following rules, where number four has the highest priority:

1. Browser default
2. External Style Sheet
3. Internal Style Sheet (inside the <head> tag)
4. Inline Style (inside HTML element)

So, an inline style (inside an HTML element) has the highest priority, which means that it will override applicable style declared inside the <head> tag, in an external style sheet, and in a browser (a default value).

Containers: Parent-Child Relationship and Inheritance

As in HTML code where tags can be nested inside other tags, the same is true of CSS rules. When rule B is positioned so as to be "sitting" inside the effects of rule A, then rule B is the **child** rule and rule A is the **parent** rule. The parent rule is also called the **container** because it contains other rules inside its area of style effects. The significance of this parent-child relationship is that, **depending on the specific properties involved** in the parent rule, styling effects from the parent container can be **inherited** by the child rule. Thus there is an inherited style effect that can occur from parent to child rules. This is called **style inheritance**. Inherited style effect is an advantage to web developers, as they can define global site style elements in the parent rules and have those styling effects be inherited by the subsequent child rules (Duckett, 2005). Its use, which saves time and effort in web page design, requires advanced CSS skills.

The Property Groups

The style properties used in CSS are grouped by type or themes. The property groups reviewed in this paper include:

Properties	Uses
Text, Font, Background	used for basic design
Margin, Padding, Border	used together to construct boxes (the box model)
Links, Position and Layout	used for advanced design elements

Only the common values for each property will be explored in this paper.

The Units of Measurement for Value Lengths

One feature of CSS that adds to its versatility is the units of length and size its values use in defining properties. Values are expressed as a % (example: 90%), or a length (example: 14px), or as a specified value (example: blue). Value lengths use the measurement units of pixels, px; points, pt; or em, which is the width of the letter m in the chosen font. Further units of measurement for value lengths include millimeters, mm; centimeters, cm; and inches, in. (Teague, 2001).

Percentages as a value for properties can be below or above 100%. Examples include: 55%, 100% (default), and 140%. The percentage is always in reference to the default dimension for the specific property being defined by the value. For example, if the Arial font has a browser default value of 12 pixels, then the value of 100% would render the Arial text at 12 pixels. A value of 200% would render the Arial text at 24 pixels.

The many types of length and size units yields to the web developer the accuracy, precision and control in their page design they have so long sought for.

All properties have a default value which is called "normal". However, the value "normal" does not have to be written in the style rule to display its default value. When a property is not specified in a CSS rule, then the browser will use its default value for that property. For example, if the color property is not included in a CSS rule, and therefore was not defined by a color, then text color will be displayed by the browser's default value of black.

Browser Conformity

Browsers are conforming more and more to CSS recommendations. Some style properties or values may not work with older browsers or with specific manufactures' versions (Schengili-Roberts, 2005). As a means to simplify this subject, the following CSS properties reviewed in this paper work consistently with **Internet Explorer 6.0** for Windows. Their style effects could possibly display differently in other browsers or older versions of Internet Explorer. The latest versions of Opera, Netscape and Mozilla, for examples, are all strictly W3C compliant. A web page constructed for Internet Explorer may not display properly through these compliant browsers (and visa versa). Programmers use special hack code to help solve browser disparity.

Furthermore, the following is not an exhaustive list of properties, but rather, a selected list of common properties. Some property groups may have more properties than what is reviewed here.

Companion Web Site

This paper has a companion web site at: <http://joepanico.com/sites/css/paper/index.html>. It provides tutorials and demonstrations of the CSS properties explained in the remaining pages of this paper. The web site was created for viewing in **Internet Explorer 6**.

Furthermore, I refer the reader to the **w3schools.org** web site for its outline of CSS properties and values, as it **is the reference for the following review of properties**.

Property Groups: A Review with Examples

Font Properties

Property	Values
font-family:	arial, verdana, tahoma, "times new roman" (fonts common to the different Operating Systems.)
font-size:	a length with units of em, pt, px, percentage, xx-small, x-small,
font-style:	italic
font-variant	small-caps
font-weight	bold
line-height	length, %
font	font-style, font-variant, font-weight, font-size/line-height, font-family (shorthand method example: {font: italic bold 16px arial;})

The following is an example of CSS rule defining font style:

```
p {font-family: veranda; font-size: 14px; font-weight: bold;}
```

Shorthand Method for Writing Property Definitions

The "font" property is a shorthand method that includes several property values to be listed without commas like this:

```
p {font: bold 125% Tahoma;}
```

The above is a shorthand method for the values of font-weight, font-size and font-family values.

Some CSS property groups include a specific property that is used as a shorthand method to reduce coding (Holzschlag, 2003). However, the shorthand method of writing rules requires a specific sequence of property values. In the case of the font shorthand, the sequence is the values for font-style, font-variant, font-weight, font-size/line-height, and font-family.

The font property **line-height** refers to the space between the bottom of one line of text and the top of the immediate line of text below.

For web demonstrations, view Fonts at the companion web site.

Text Properties

Property	Values
color	color
letter-spacing	length
text-align	left, right, center, justify
text-decoration	underline, overline, line-through,
text-indent	length, %
text-transform	capitalize, uppercase, lowercase
word-spacing	length

The property "letter-spacing" refers to the space between letters, as "word-spacing" refers to the space between words. Here is a CSS rule using some of the text properties:

```
p {font: 14px verdana; color: blue; letter-spacing: 2px; word-spacing: 4px;}
```

Another CSS rule might include:

```
p {font: bold 90% arial; color: beige; line-height: 8px; text-decoration: underline;}
```

For web demonstrations, view Text at the companion web site.

Class Selectors

Though class selectors are not a property group, it is appropriate to explain this category of selectors before providing further rule examples.

The above examples of CSS rules where applied to the p command for the <p> HTML tag. When a CSS rule is applied to a HTML command, the command is called a selector. CSS rules can be applied to many HTML commands. Moreover, webmasters can create their own selectors (Teague, 2001). Some of the most common HTML commands that CSS are applied to include:

p, h1-h7, body, ol, ul, li, a, table, and td

These commands are used to format text and are the most commonly used when constructing web pages in general. However, webmasters need more style rules than few HTML commands listed above afford. Web authors can create their own selectors called **class selectors**. A class selector can be given any name, but it is prudent to use a name that is an abbreviated description of its style definition. Here is an example of a class selector named "redtext":

```
.redtext {font-family: arial; font-size: 14px; color: red; text-indent: 10px;}
```

In this rule the name .redtext was used in place of using an HTML command as selector. A period (.) always precedes a class selector name. Class selectors can be applied to the web page code using the or <div> tags like this:

```
<span class="redtext"> Four score and seven years ago our fathers brought.... </span>
```

Or by:

```
<div class="redtext"> Four score and seven years ago our fathers brought.... </div>
```

For web demonstrations, view Class Selectors at the companion web site.

Background Properties

Properties	Values
background-color	color name, color-rgb, color-hex
background-image	url
background-repeat	repeat, repeat-x, repeat-y, no-repeat
background-attachment	scroll, fixed
background-position	top left, top center, top right, center left, center center, center right, bottom left, bottom center, bottom right, x-%, y-%, x-position, y-position
background	background-color, background-image, background-repeat, background-attachment, background-position (shortcut ex.: background: url('image.gif') no-repeat fixed;)

CSS provides flexibility for background style. A background image can be defined to tile (repeat) only vertically against the left browser border, or horizontally against the top browser border. Or, background images can tile at any position across the page. Background images can be displayed only once with no tiling effect, and as such, can be placed in various positions on a web page. They can behave as a normal image and roll up the page, as does other page content, when the viewer is scrolling down. Or, background images can remain in a fixed position and not roll up or down during page scrolling (W3 Schools, 1999).

A CSS rule to have the background image tile only horizontally on the x-axis of the page could be written like this:

```
body {background-image: url('bg-1.jpg'); repeat: repeat-x;}
```

A CSS rule could be written as the following to have a background image positioned in the center of the page and not roll up or down (fixed) when the viewer is scrolling:

```
body {background-image: url('bg-2.jpg'); repeat: no-repeat; background-attachment: fixed; background-position: center center;}
```

For web demonstrations, view Background at the companion web site.

The Box Property Groups – Margins, Padding and Borders

The three groups of properties - margins, padding and borders - can be used together to create the box model, or box effect on web pages (Schengili-Roberts, 2005). Boxes, like tables, can be visible or invisible, with text or other objects inside. Boxes are very versatile in page layout design.

Margin Properties

Properties	Values
margin-top	length, %
margin-right	length, %
margin-bottom	length, %
margin-left	length, %
margin	(shorthand for top, right, bottom, left, example: p {margin: 100px 200px 200px 50px;} or p {margin: 20px;} The 20px defines all four sides.)

Margin properties are a means to apply a "cushion" of empty space around the four sides of any object or element on a web page. Margins are often placed around text (in much the same way as a word processor has four margins on a page).

Margins provide space from the outside perimeter of a box in toward the element. Padding (see next property) provides empty space from the inner element outward toward the margin space. Margins are empty spaces on the outside of both the element and the element's padding (Schengili-Roberts, 2005). Margins do not display their own background properties.

The margin property uses the shorthand method in the sequence of top, right, bottom and left. An example of a margin around text could be:

```
.blue-text (font: 12pt courier; color: blue; margin: 10px 20px 10px 20px;)
```

Margin Flow

Margins exist in positions on web pages relative to (offset from) the margins and positions of their neighboring elements. The way elements on a page position themselves relative to each other is called **flow** (Dunkett, 2005). If text styled with a margin is in the body of the web page (that is, if the text's parent container is the entire web page) then its left margin will be positioned from the left side of the browser window. In contrast, if the same text exists inside a parent text box then it is a child to the parent box. As such, the text's left margin will be positioned relative to (or offset from) the margins of the parent box. The margin of the child text box will begin from the nearest margin boundaries of the parent text box. Margins are position-sensitive to the margins and positioning of neighboring objects and elements. The term flow refers to this position-sensitivity of margins (and other style properties, as well).

Width and Height

Before proceeding to padding properties it is appropriate at this point to introduce two dimension properties: **width** and **height**. These two properties are often included in the box definitions. Here is an example:

```
.box-1 {margin: 20px; font: 24px bookman; width: 300px; height: 300px;}
```

Padding Properties

Properties	Values
padding-top	length, %
padding-right	length, %
padding-bottom	length, %
padding	padding-top, padding-right, padding-bottom, padding-left (shorthand example: .box {padding: 10px 10px 10px 10px; })

Padding, as mentioned above, provides an empty space immediately surrounding the element it is applied to (often text). Margins define a second layer of empty space around the padding space. One difference between margin space and padding space is that the outer perimeter of the padding space determines the position of the border properties (see next property) (Schengili-Roberts, 2005). Furthermore, background color and images fill the padding space, but not the margin space. Margin space assumes the background of its parent container. Here is an example of defining padding around text:

```
.text {font: 16pt comic sans ms; color: blue; padding-top: 20px; padding-bottom: 20px; margin-left: 30px;}
```

In this rule the margin-left will offset the text 30 pixels toward the right, and the padding will add extra "cushion" of space, 20 pixels, to the top and bottom the text.

Padding properties have two versions of shorthand:

.text {padding: 10px 20px 10px 20%;} Or, just one value to represent all four sides:
.text {padding: 20px;}

Border Properties

Properties	Values
border-width	thin, medium, thick, length
border-style	hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
border-color	color name, hex color, RGB color
border	border-width, border-style, border-color (shorthand example: .box {border: thin solid red;})

Border properties apply a borderline (a stroke) around an element or around its padding (not around the margin). Most typically used around text, borders can be defined in any color, and in several styles. Used with margins, padding and background properties, border styles are a part of the **box model effect**.

Text with a border around it may be defined like this:

.text-border {font: 16pt verdana; color: red; margin-left: 80px; padding: 15px; border: 1px dashed blue;}

The above text class rule is also a box. Here is another example using background properties with a box:

.box-1 {font: bold 12px verdana; color: red; margin-left: 100px; padding: 10px 30px 10px 30px; background-image: url('bg-1.gif'); background-repeat: repeat-y; border: 2px double #ffccff;}

For web demonstrations, view Box Model at the companion web site.

Link (Anchor) Styles

There is only one anchor command in HTML; the <a> tag. However, the anchor tag has four functions that require it, as it were, to use four sub-selectors or **pseudo classes** as they are called. A hyperlink can be in one of any four states: unvisited, visited, hovered over with the mouse, and in the instant of being click on (Teague, 2001).

Pseudo Class	State
a:link	selector for unvisited links.
a:visited	selector for visited links.
a:hover	selector for hovering over a link.
a:active	selector for clicking on a link.

Here are examples of defining the style properties for the four states of the anchor selector:

```
a:link {font-family: verdana; font-size: 12px; text-decoration:none; color: blue;}
a:visited {font-family: verdana; font-size: 12px; text-decoration:none; color: purple;}
a:hover {font-family: verdana; font-size: 12px; text-decoration:underline; color: red;}
a:active {font-family: verdana; font-size: 12px; text-decoration:none; color: blue;}
```

In these examples the unvisited link is blue, visited is purple, hover is red and underlined, and when clicking on the link, it is blue.

For web demonstrations, view Links at the companion web site.

Position Properties

Property	Values
position	absolute, relative
top	length, %
left	length, %
bottom	length, %
right	length, %
z-index	number

The position properties allow a webmaster to place objects anywhere on a page with pixel accuracy. There are two common types of position values: **absolute** and **relative**. In both types of positioning the element being positioned will position itself in context to whatever parent element (container) it is in. The **absolute** value ignores all other content contained in the parent element, and thereby, positions the object it styles exactly by the top, left, bottom and right coordinates of the parent container. The absolute value can be used to purposely layer one object on top of another.

In contrast, a **relative** value for positioning will position an object inside the parent container relative to the nearest elements that are also in the parent container. Relative positioning, like margins, are as though "sensitive to" the margins and positions of neighboring elements in the container, and thereby, offset their position to **flow** accordingly (Duckett, 2005). The relative value to the position property affects the flow characteristics of the object it defines. An element with a relative position value will not overlap on top of another element. Relative positioning takes in to account the positions of the other elements in the container. Examples follow:

```
.text {font: 14px verdana; position: absolute; top: 200px; left: 100px;}
```

This example is a style rule whereby the text it defines will take an absolute position within its parent element (the page body or a box, for examples) even if there is already page content in that position. Two elements could overlap if not placed carefully.

```
.text {font: 14px verdana; position: relative; top: 200px; left: 100px;}
```

In this text rule the positioning is relative, therefore, regardless of which parent container this text is inside (the body or a box) it will offset itself to take a position 200 pixels below the element directly above it, and 100 pixels right of the element to its left. In this manner the position is not absolute, but relative in reference to all the other elements in the parent container.

The z-index

If two objects are intended to overlap, the z-index is used to determine which object will be positioned on top of the other (Teague). The values of the z-index are any whole integer (positive or negative value). The object with the higher z-index number will be positioned on top of the object with a lower number. An object defined with a z-index of 2 will position itself on top of an object defined with a z-index value of 1. The z-index defaults with value 1.

For web demonstrations, view Position at the companion web site.

Layout Without Tables

All the CSS property groups reviewed above are combined in creative ways to make attractive web sites. Web developers configure the layout of page elements by three methods innate to CSS properties: **flow**, **floating**, and **positioning** (Duckett, 2005).

Layout by **flow** recognizes the natural placement of elements one after another that occurs in the order that the elements are written in the source code. Each element is sequentially placed vertically down the page.

Layout by **floating** involves using the **float property** that gives a left or right value to the objects it defines. Objects with a "**float: left;**" style will align to the left side of the page layout. Objects with a "**float: right;**" style will align to the right side of the page layout.

Layout by **positioning** will position objects in an absolute or relative value exactly where the web author intends, as described above. **Note that unfortunately this property is not completely supported by Internet Explorer 6.**

For web demonstrations, view Layout at the companion web site.

Conclusion

CSS is the new way to make professional web pages. Its versatility and depth of creative potential will engage web developers for many years to come. Webmasters now are not limited by their tools, but rather, are free to explore web design as far as their imagination and skills can carry them.

CSS version 3 will be recommended soon. Browser manufactures are still adjusting their software features to comply with the W3C recommendations. Our Web presence on the computer screen and on mobile and multimedia devices will evolve to levels of technological advancement we cannot image. While offering exciting possibilities today, CSS will be an integral part of the larger picture of the World Wide Web and global communication systems of the future.

Appendix A

Joe Panico

U06D1 Paper Proposal

Written 4-16-05

A Historical Review of the World Wide Web and the Emergence of CSS

Background

HTML has evolved quickly since the founding of the World Wide Web Consortium (W3C) in 1993. In the process there were many frustrations and conflicts that bore heavily on the web development community. HTML (Hypertext Markup Language) was never intended to be a language for web design or aesthetic appeal. It is a markup language that provides structure to web pages. The overzealous browser manufacturers and the public's passion to acquire web skills forced HTML to be used in a manner it was not designed to do: provide creative presentation. Through long patience the W3C (the governing body for the Web) eventually took center stage and its recommendation for a style language is finally gaining public recognition. Cascading Style Sheets (CSS) is the actual recommendation that the custodians of the World Wide Web have offered the public as the means to create stunning, versatile and professional looking web pages.

Web developers have long been stymied by the limitations of HTML and are now waking up to the call of the W3C. Public frenzy dominated the beginning of the Web. Now a code of standards-based professionalism is grasping the web community and it is slowly centering web page development around CSS as it should.

Intention

My paper, in part, is intended to provide a review of the emergence of the Web, the historical context of web page language development and the eventual adaptation of Cascading Style Sheets as the standard of web page construction and design.

Web language development did not occur in a vacuum. My paper will show how the Web craze temporarily affected the emergence of CSS, the complement and extension of HTML. I will cover the limitations of HTML and the advantages of CSS. The remainder of the paper will then review the syntax, and rules of CSS for various design elements of web page construction. I will provide links to online example pages of these CSS effects.

Preparation, Research and Timeline

My initial research to date has astounded me as to the depth and complexity of the subject. The more I research the more I find there is to know and report. I have found the paper's introduction and historical review is lengthier than I expected. In that sense I find the paper taking on its own dimension. Therefore there may be less review of CSS rules than in my initial outline, but I feel this proportion of historical review to CSS examples will be balanced and appropriate.

Further consideration is speed of progress. I am spending more time than expected on the history and background. My manner of research is painstaking: challenging, but slow. However, I feel that with persistent effort I will finish on schedule.

What is fascinating for me is that as I review (and learn) the CSS rules I can, if time allows, then create pages using those rules and post them on the web as demonstrations of CSS. Though this will benefit me as a webmaster, it is not a requirement of the final paper and so will I will only extend my efforts in this manner if it is convenient to do so.

A progress report of my paper will be ready by May 8th. The rough draft of the final work will be ready by May 15th. And, the Final Paper will be submitted on May 22nd.

Conclusion

The advent of the Web was a global event that spread with the speed of instantaneous communication. It is comprised of many industries all of which are struggling to keep up with the rapid pace of their own technological evolution. If any area of the Web struggled with establishing professional standards it was web page development. Today web developers are able to look back at their young history and realize as a profession that the W3C recommendations are the most prudent way to proceed into the future. CSS is at the heart of those recommendations, providing a means of elegant expression, and a universal accessibility of Web content to the world of Web technology including mobile devices.

Appendix B

Joe Panico

U09d1 – Progress Report

Written May 2, 2005

Cascading Style Sheets: A History, Review and Demonstration

Introduction

As I approach the remaining 3 weeks to the due date of our final project, I am relieved to see the paper take a form that I will be proud to present to the class. Cascading Style Sheets (CSS) is a style language for web pages that is still new to many webmasters. I chose this topic so I may learn the language and I am pleased with my progress thus far.

My paper will provide a review of the beginning of the World Wide Web, HTML and the emergence of CSS as a compliment to, and a solution for the limitations of HTML. The paper will then review the CSS language: its syntax, and properties groups. I will provide examples style rules. Of equal importance, I am constructing web pages that demonstrate these style effects. For without such demonstration, that is, seeing the style properties applied to page design, an explanation of CSS rules alone is almost meaningless.

Hence my project has three components: history, CSS rules, and demonstration web pages.

Progress

The history portion of my paper is quite complete and will need no further editing. The CSS review portion is 85% complete. The web page demonstrations are 60% complete. Web page construction is time consuming. If I were not creating an accompanying web site I would have been nearly done with my paper by now. However, having to demonstrate CSS forces me to learn the language more thoroughly, which is my very purpose in choosing the topic.

As I explain CSS properties I then use those style effects in the construction of the succeeding demonstration pages. Therefore the demonstration pages have an increasingly more stylish look as the "lessons" progress.

Work Remaining

As the rough draft of our papers will be posted on the 11th or 12th of next week I am going to work diligently on the paper to have it as close to 100% finish by then as possible. Remaining work includes showing how all the CSS properties combine to create web page layouts. I still need a table of contents, references list, and I need to carefully cite all referenced content. After submitting the rough draft I will then complete the web page demonstrations.

Our papers' final deadline is Sunday, May 22nd. I feel confident that after posting the rough draft that I will have enough time (literally 10 days) to refine the final draft and complete the web pages.

Problems

The problem is stamina. I have been ill and have fallen behind my schedule. Furthermore, I lost a web page that I spent three days constructing; lost time! However, the time I have spent authoring web pages demonstrating CSS has improved my ability to write my paper. I have more confidence. My method of operation is to pace myself and continue tirelessly, persistently, until I am done. I feel I will complete the work on time.

Conclusion

This project is demanding. However, it is worth every effort, for when I am finished I will be able to construct web sites using CSS. This will be a long awaited and satisfying accomplishment.

References

Berners-Lee, T. (1998). Retrieved April 7, 2005, from <http://www.w3.org/People/Berners-Lee/FAQ#InternetWeb>.

Duckett, Jon (2005). *Accessible XHTML and CSS web sites*. Indianapolis, IN: Wiley Publishing.

Griffin, S. (2000). *Internet pioneers*. Retrieved April 7, 2005, from <http://www.ibiblio.org/pioneers/lee.html>.

Holzschalg, M. (2003). *Cascading style sheets: The designer's edge*. Alameda, CA: Sybex.

Schengili-Roberts, K. (2004). *Core CSS: Cascading style sheets*. Upper Saddle, NJ: Pearson Education.

Shafer, D. (2003). *HTML utopia: Designing without tables*. Melbourne, Australia: SitePoint.

Teague, Jason C. (2001) *.DHTML and CSS for the world wide web*. Berkeley, CA: Peachpit Press.

WaSP (1998). Retrieved April 22, 2005 from <http://webstandards.org>.

W3 Schools (1999). *CSS tutorials*. Retrieved April 26, 2005, from http://www.w3schools.com/css/css_intro.asp.