

# Cascading Style Sheets (CSS)

# What Web Designers Need to Know About CSS

- Page Layouts
- Styling and Making
- Box model and Po
- Typograph
- Styling Fo
- Usir
- Best
- CSS Hack

**CSSTidy**

**What is Elements?**

help designers write CSS  
at a framework, it's its own

**Vertical align div**

One of current CSS left outs is vertical align div. And before CSS 3 comes we have to use some tricks to solve this problem. I looked over for some solutions and it all comes up to defining it

margin: 10px 20px;

margin: 10px;

**HEADLINE EXAMPLE**

**LOREM IPSUM DOLOR SIT AMET**

ut labore et dolore magna aliqua. Ut enim

nisi ut aliquip ex ea commodo consequat.

Is

- Adds style and usability to external links with icons
- Can be easily uploaded to your [website hosting](#) server
- Free of course

No

# Early HTML version 3.2 example

```
<font face="Arial, sans-serif" size=2 color=blue>  
Chapter One </font>
```

```
<p><font face="Times Roman, serif" size=1  
color=black> The mean mode or ... </font></p>
```

```
<i> This text is emphasized. </i>
```

```
<i><font color=red> This is emphasized in  
red.</font></i>
```

```
<i> <font color=blue> Some blue . </font></i>
```

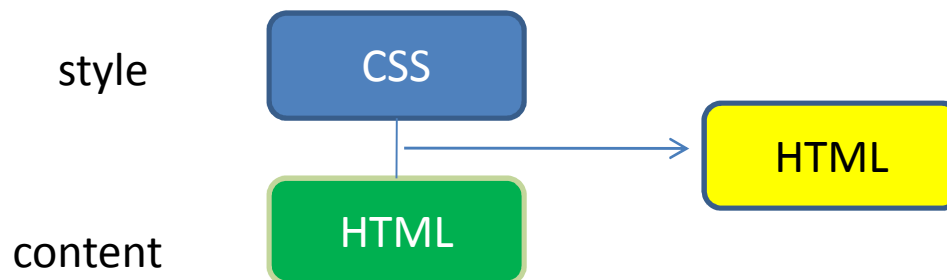
presentation                      content

# HTML version 3.2 issues

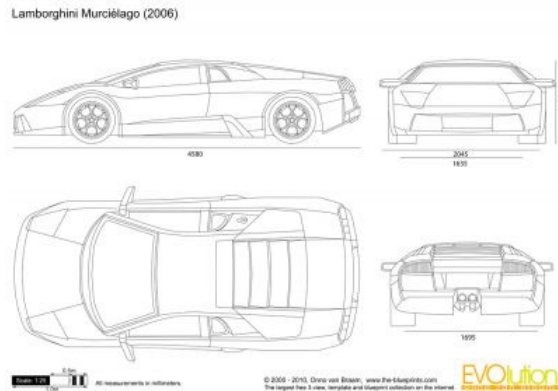
- A problem with early HTML web pages (prior to CSS) is that the **content** and its **presentation** are closely integrated
- The **meaning of content** and **how that content is presented** should be separated. Why?
- The “user agent” receiving the HTML may be a desktop **browser**, a **printer**, a **PDA**, a **cell-phone**, or a **tablet** – the HTML intended for display to the browser may not be acceptable on the printer or other devices
- Also, you may want to see the web page using your **own preferred presentation style** as defined in the browser
- The process **of updating the appearance of the content** should be simple and efficient

# Style Sheets

- With HTML v4 (published 1998) a new approach to web page definition was developed using **style sheets**
- A style sheet is a set of defined presentation instructions that is separate from the content
- The concept of style sheets had been around since SGML days (1980s)



# Style and Content Together



+



=



# The creator of CSS came from the land that gave the world ...



Håkon Lie – developer  
of CSS (1994)

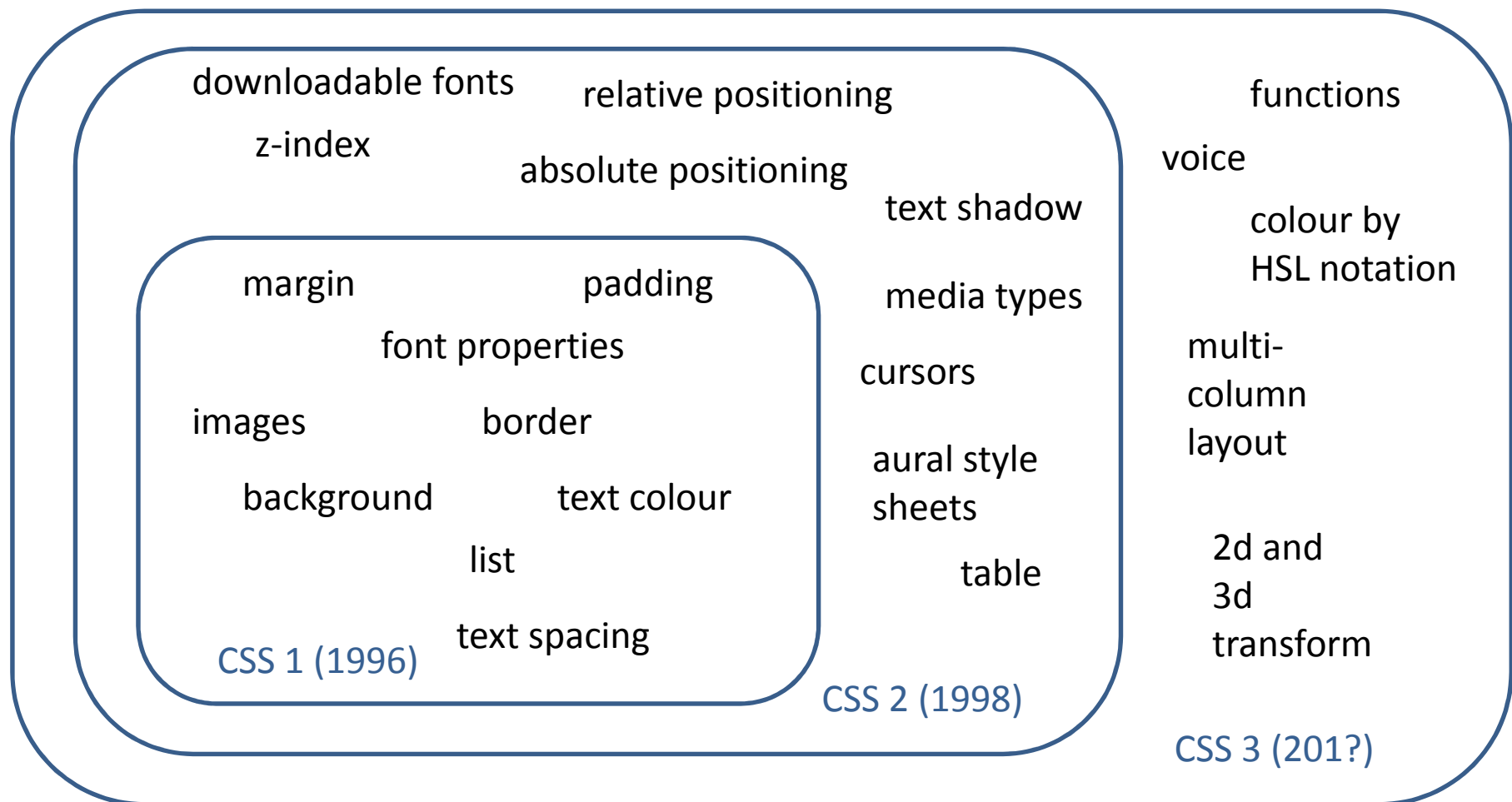


# Origin of CSS

- W3C has produced three style recommendations CSS1, CSS2, and CSS3 – each level builds on the previous version - <http://www.w3.org/TR/CSS21/>
- Not all browsers implement these changes consistently
  - <http://tools.css3.info/selectors-test/test.html>
- Early browsers (MS Internet Explorer 3 and 4) did not fully support CSS
- As of July 2010, no browser has yet fully implemented CSS3 – some more than others
- [http://en.wikipedia.org/wiki/Comparison\\_of\\_layout\\_engines\\_\(CSS\)](http://en.wikipedia.org/wiki/Comparison_of_layout_engines_(CSS))



# Evolution of CSS Features from W3C



# Why CSS?

- CSS skills are essential for web page design
- HTML skill allows you to understand how to structure the HTML content but not **how to present it effectively**
- Easily change the presentation of an entire web site by modifying a single CSS style sheet
- CSS skills vital in web projects
- Use a CSS validator to check your CSS is correct
- If there is a problem with your CSS, usually there are no error messages – check with Firebug on Firefox or Internet Tools on IE (ver 8)

# What is CSS ?

- Like HTML CSS is simple **human-readable** text
- CSS is not a programming language like Java or PHP
- CSS is not the same as HTML – CSS cannot be present without HTML or XSL
- CSS will help simplify your web content presentation and make it more manageable
- CSS will also work with XSL, another web technology

# CSS Style Attribute and Tag

- CSS has a **style attribute** and **style tag** (selector)

`<h1 style = "property : value;  
property : value; ... ">`

declaration

`<style>  
selector { property : value;  
property : value; ... }  
</style>`

the selector is usually an HTML tag name  
The final **;** is optional.

# CSS Style Layout

- CSS definitions are written **free format**

```
<style type="text/css">
```

```
h1 { color: blue; }
```

```
h2 { color: green;  
    }
```

```
h3 { color: red;  
    text-weight: normal;  
    }
```

```
</style>
```

The closing brace  
can go anywhere  
and each declaration  
can be on a  
separate line.  
Optimizing the CSS  
for **human  
readability**  
is desired.

The order of the CSS properties listed  
in the style does not matter to the user agent.  
If properties are duplicated in the same style,  
the last one defined is used.

# CSS Properties

- Properties are relevant to the selector
- Properties and values are case insensitive but standard is to use lowercase
- Some of the CSS property names are not consistent unfortunately: e.g. `color:blue` ✓  
**not** `text-color:blue` ✗ **or** `text:blue` ✗
- Examples of CSS property names: background, border, margin, padding, font-size, font-family, word-spacing, visibility
- Not every property works consistently for each browser! Test for each browser and version.

# CSS Values

- Values can be numbers, strings, keywords, lengths, colour values, urls or percentages
- For numbers, only decimal values
- Strings, use double or single quotes
- For keywords (e.g. *auto*, *none* or any of the known colour names), do not use quotes (e.g. `color: "red"` is illegal; `color:red` is legal) ✓
- A zero length value does not require a length identifier (e.g. CSS style `margin:0` )
- Colour values can be a keyword or RGB notation (or HLS notation for CSS 3 browsers)



# CSS Strings

- A string can be written with **double quotes** or **single quotes** (no mixing though)
- A **double quotes** can appear inside a double quoted string but the **double quotes** is **escaped** by preceding it with a **backslash**. Similarly for **single quotes**

`"here is my 'string' "`

`"this is another \"string\""`

`'another "string" '`

`'yet another \'string\' '`

# Colour keywords

- colour keywords: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, and yellow.

<b>maroon</b> #800000	<b>red</b> #ff0000	<b>orange</b> #ffa500	<b>yellow</b> #ffff00	<b>olive</b> #808000
<b>purple</b> #800080	<b>fuchsia</b> #ff00ff	<b>white</b> #ffffff	<b>lime</b> #00ff00	<b>green</b> #008000
<b>navy</b> #000080	<b>blue</b> #0000ff	<b>aqua</b> #00ffff	<b>teal</b> #008080	
<b>black</b> #000000	<b>silver</b> #c0c0c0	<b>gray</b> #808080		

**RR GG BB**

is the format for mixing red/green/blue.

If the two hexadecimal digits of the colour are the same, then you can specify the one digit.

e.g. white is #fff or #ffffff

alternative **R****G****B** formats:

rgb(255, 255, 255) or

rgb(100%, 100%, 100%) = white

<http://www.w3.org/TR/SVG/types.html#ColorKeywords>

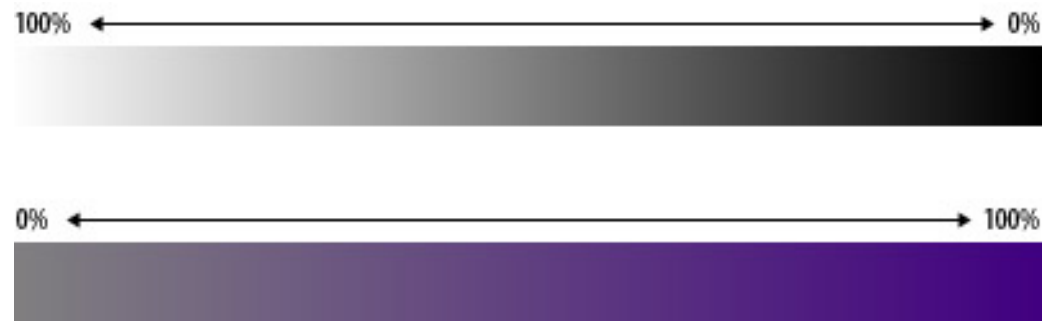
# HLS Notation

- CSS 3 introduced the HLS colour notation
- H = hue L = lightness S = saturation
- **R****G****B** colour notation is hardware-oriented (based on CRT monitors)
- **R****G****B** is not intuitive beyond the basic colours
- Hue is the **angle** of the colour circle
- 0% lightness is black; 100% lightness is white
- 100% is full saturation; 0% saturation is grey

# HLS Notation

- Examples,

```
b { color: hsl(0, 100%, 50%) } /* red */  
p { color: hsl(120, 100%, 50%) } /* green */  
div { color: hsl(240, 100%, 50%) } /* blue */
```



# Transparency – Alpha channel

- Another CSS 3 technique used by FF3+, Opera, Chrome, Safari) is using the **alpha** channel in RGB or HSL

```
background-color: rgba(100, 155, 164, 0.4);  
/* alpha=0.4*/
```

```
background-color: hsla(200, 40%, 20%, 0.4);
```

Using this method the settings do not affect child elements as **opacity** property does.

# Vendor-specific extensions

- Identifiers may start with a hyphen (-) or underscore (\_) ... they are reserved for vendor-specific extensions
- E.g. **-moz-box-sizing** for Mozilla browsers  
**-ms-**, **ms-** for Microsoft browsers  
**-moz** for Mozilla  
**-o-** for Opera  
**-rim-** for Research in Motion  
**-webkit-** for Apple

# CSS Values - Examples

```
<style type="text/css">
  p { background-color: gray;
        background-position: 40% 50%;
        background-size: 10em 10em;
        background-image: url(blue.png);
        background-repeat: round round;
        border-style: solid;
        border-color: #FF30A0;
        width: 50px;
        font-family: 'Courier New Times' ;
      }
</style>
```

Annotations:

- ← CSS colour keyword (points to `gray`)
- ← CSS percentages (points to `40% 50%`)
- ← CSS lengths (points to `10em 10em`)
- ← CSS URL (points to `url(blue.png)`)
- ← CSS keywords (points to `round round`)
- ← Colour in RGB format (points to `#FF30A0`)
- ← CSS length (points to `50px`)
- ← CSS string (points to `'Courier New Times'`)



# CSS errors

- If the CSS property is **unknown**, the user agent ignores it  
h1 { color: blue; **angle: 30deg;** }  
is treated as h1 { color: blue; }
- If the CSS declaration uses an **illegal value**, the user agent ignores it  
img { float: left **here** } is ignored
- If the CSS declaration is **malformed**, the user agent tries to work around it  
p { color: blue; **color** } uses p { color: blue }

# Lengths

- Fonts scaled using the **font-size** property
- Two types of length units: relative and absolute
  - Relative lengths units specify a length relative to another length property – also called ‘elastic’ or ‘fluid’
    - More easily scaled from one display device to another
    - **em** – the ‘font-size’ of the relevant font
    - **ex** – the ‘x-height’ of the relevant font (rarely used)
    - **px** – pixels, relative to the viewing device

# Lengths

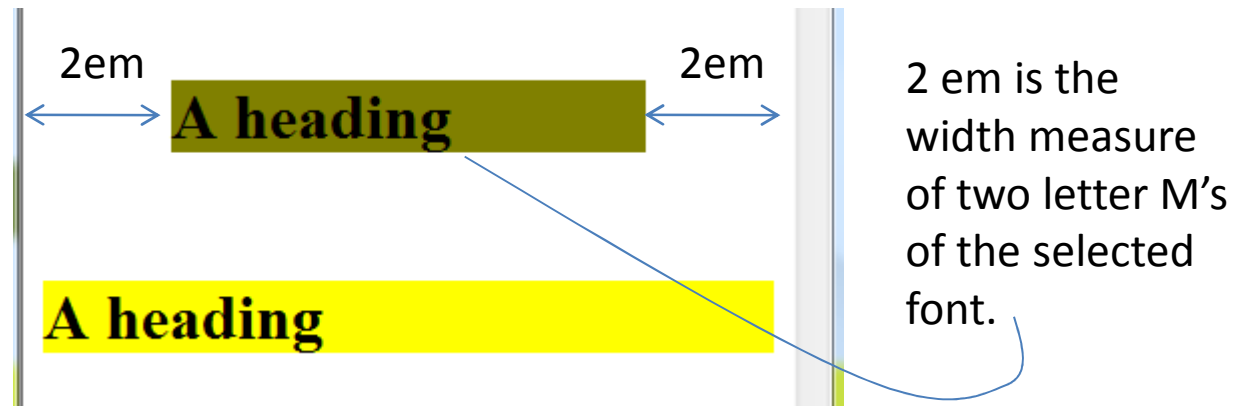
– Absolute lengths units are useful only when the physical properties of the output device (e.g. printer) are known – do not use for video display

- **in** – inches
- **cm** – centimeters
- **mm** – millimeters
- **pt** – points – 72 points equal one inch – only for print
- **pc** – picas – 1 pica equals 12 points—only for print

**1em = 12pt = 16px = 100%**  
(generally by default  
unless you change the  
working font )

# Lengths CSS

`<h1 style="margin:2em;">` defines a margin space around the h1 heading. The space is twice the length of the font's lower-case m as a heading.



# CSS Lengths - Example

- Define four boxes in CSS – each with a different margin: 1em, 2em, 10px, and 40 px

```
<style type="text/css">
  .box1 { margin:1em;
          background-color:yellow;
        }
  .box2 { margin:2em;
          background-color:red;
        }
  .box3 { margin: 10px;
          background-color:green;
        }
  .box4 { margin: 40px;
          background-color:orange;
        }

</style>
</head>
<body style="border:solid 1px black">
<p class="box1">
The quick brown fox jumped over the lazy dog.
</p>
<p class="box2">
The quick brown fox jumped over the lazy dog.
</p>
<p class="box3">
The quick brown fox jumped over the lazy dog.
</p>
<p class="box4">
The quick brown fox jumped over the lazy dog.
</p>
```

# CSS Lengths - Example

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

Repeat above with Arial font 24px font size.

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

The quick brown fox jumped over the lazy dog.

The diagram illustrates the effect of different CSS length units on text boxes. The top four boxes (yellow, red, green, orange) are of the same height but have different margins, demonstrating the effect of different 'm' widths. The bottom four boxes (yellow, red, green, orange) are of the same height and have the same margins, demonstrating the effect of the same 'px' (pixel) width.

The yellow and red boxes have different margin widths because the font used has a different "m" width.

But the green and orange boxes use the same margin widths because the CSS style used is "px" = pixel.

# Fonts in CSS

- A font is a typeface used to display text
- A font is an operating system resource, not a browser resource
- CSS recognizes five *generic font families*: serif, sans-serif, monospace, cursive, and fantasy
- Monospace is a fixed size font; serif and sans-serif are typically proportionally spaced fonts



- **Serif** fonts have flared or tapered ends to the letters; sans-serif fonts do not.

AaBbCc    AaBbCc    AaBbCc



# Fonts in CSS

- **Serif** family of fonts include: Times New Roman, Century Schoolbook and Garamond
  - Most widely used of the font families for print books, magazines
  - But sans-serif considered easier to read on computer screens

<http://www.w3.org/TR/CSS2/fonts.html#q1>

# Fonts in CSS

- **Sans-serif** font family includes: Arial and Verdana
  - Much easier to read on monitors as the “serifs” can blur on lower resolution screens
  - Verdana was invented for use on the web
- **Monospace**: Courier New, Lucida, Consolas
  - Also called “fixed pitched” fonts because all characters have the same width
  - Use for showing programming code or implying typewritten text
  - Before CSS this was required for text alignment in table format especially for columns of dollar amounts (decimal point alignment)
- [http://en.wikipedia.org/wiki/Samples\\_of\\_Monospaced\\_typefaces](http://en.wikipedia.org/wiki/Samples_of_Monospaced_typefaces)

# Fonts in CSS

- Limit usage of fantasy and cursive fonts as they are not widely available and can be hard to read – especially for non-English speakers
  - If you must use them, keep text very short
- **Cursive** e.g. **Comic Sans**, *Vivaldi*
- **Fantasy** e.g. **COPPERPLATE**, **Impact**

# Fonts in CSS

- When specifying the font family, list your preference first, and the most generic last
- If the font name has multiple words, put it in double quotes, or in single quotes if using the style attribute

```
p { font-family: "Courier New", Courier, monospace }
```

```
h1 { font-family: Verdana, Geneva, Arial, sans-serif }
```

```
h2 { font-family: "Courier New", Courier, monospace }
```

```
h3 { font-family: "Times New Roman", Times, serif }
```

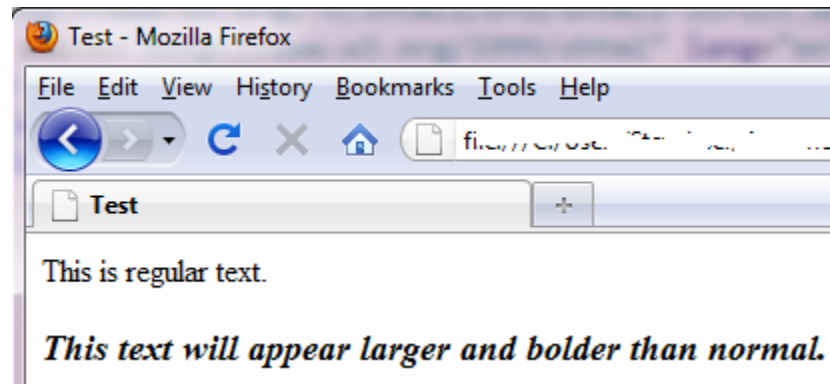
# Fonts in CSS

- **font-weight** sets the level of 'thickness' of the characters in text
  - Default is `normal` (or use number 400)
  - `bold`, `bolder`, `lighter` same as values 700, 800 and 300
- **font-style** indicates the style
  - Default is `normal`
  - *italic*, *oblique*
- **font-size** sets the size of the font
  - Can use a length value (e.g. 12px or 1.2em or 120%)

# CSS Fonts Example

```
<style type="text/css">
  p    { font-family: "Times New Roman", Times, serif;
        font-weight: 800;
        font-style: oblique;
        font-size: 18px;
        }

</style>
</head>
<body>
This is regular text.
<p>This text will appear larger and bolder than normal.
</p>
.. ..
```



# CSS Text Properties

- `line-height` refers to amount of space between lines of text
- `text-decoration` can be set to normal, underline, line-through, or none
- `text-transform` can be none, capitalize, uppercase or lowercase
- `text-align` can be left, right, center, justify
- `text-indent` is amount of indented space



# Three paragraph examples

```
<p style="line-height:130%; text-decoration:line-through; text-align:right">  
"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor  
</p>
```

```
<p style="text-transform:uppercase; text-align:justify; text-indent: 100px;">  
"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor  
</p>
```

```
<p style="line-height:160%; text-decoration:underline; text-align:center;">  
"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor  
</p>
```

~~"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."~~

"LOREM IPSUM DOLOR SIT AMET, CONSECTETUR ADIPISICING ELIT, SED DO EIUSMOD TEMPOR INCIDIDUNT UT LABORE ET DOLORE MAGNA ALIQUA. UT ENIM AD MINIM VENIAM, QUIS NOSTRUD EXERCITATION ULLAMCO LABORIS NISI UT ALIQUIP EX EA COMMODO CONSEQUAT. DUIS AUTE IRURE DOLOR IN REPREHENDERIT IN VOLUPTATE VELIT ESSE CILLUM DOLORE EU FUGIAT NULLA PARIATUR. EXCEPTEUR SINT OCCAECAT CUPIDATAT NON PROIDENT, SUNT IN CULPA QUI OFFICIA DESERUNT MOLLIT ANIM ID EST LABORUM."

"Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."

# Fonts and the Web

- Can a custom font be used in a web page?
- CSS level 2 introduced the @font-face property to allow the HTML to access different fonts
- Extra time needed to download the font file
- Font technologies: Embedded OpenType (EOT) – only Microsoft IE 4+, sIFR (Scalable Inman Flash Replacement), Font Linking
  - See <http://www.fontembedding.com/>
  - <http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/cssFontDownloadTest.html>

# CSS Property Shortcut

- For defining background, border, font, margin, or padding properties, there is a shortcut to collapse all the sub-properties into one.
- Order of the values does not matter

```
style="font-weight: bold; font-family: Verdana, sans-serif; font-size: 2em; line-height: 1.2em;"
```


*can be collapsed into this CSS shortcut:*

```
style="font: bold 2em/1.2em Verdana, sans-serif; "
```

# CSS Property Shortcut

- Another example:

```
style="padding-top:5px; padding-right:10px; padding-bottom:8px; padding-left:12px;"
```



```
style = "padding: 5px 10px 8px 12px;"
```

The order is : **T** **R** **B** **L** (clockwise from noon) - or **T****R**ou**B**Le

style = "padding: 5px 10px;" means 5px for top and bottom; 10px for left and right

style = "padding: 5px;" means 5px all around

# Opacity

- Visual elements on a web page can be made transparent with the CSS `opacity` property
- **Non**- IE browsers:



`opacity: 1.0; /* opaque */`

`opacity: 0.7;`

`opacity: 0.5;`

`opacity: 0.2;`

`opacity: 0.0; /* transparent */`

# Opacity

- IE browsers ( IE v6-8) use the **filter** property if the element uses the **hasLayout** property (e.g. table, img, input, button, textarea)

**filter**: alpha(opacity=40);

For IE 8 use the **-ms-filter** property:

**-ms-filter**:

“progid:DXImageTransform.Microsoft.Alpha (opacity=40)”;

To work in all IE versions, define the **filter** property after the **-ms-filter** property.

# URL value

- URLs to image files are values to CSS properties – use the *url* CSS function
- URL may be single-quoted or double-quoted or not quoted

```
body { background:  
    url("http://www.example.com/forest.jpg") }
```

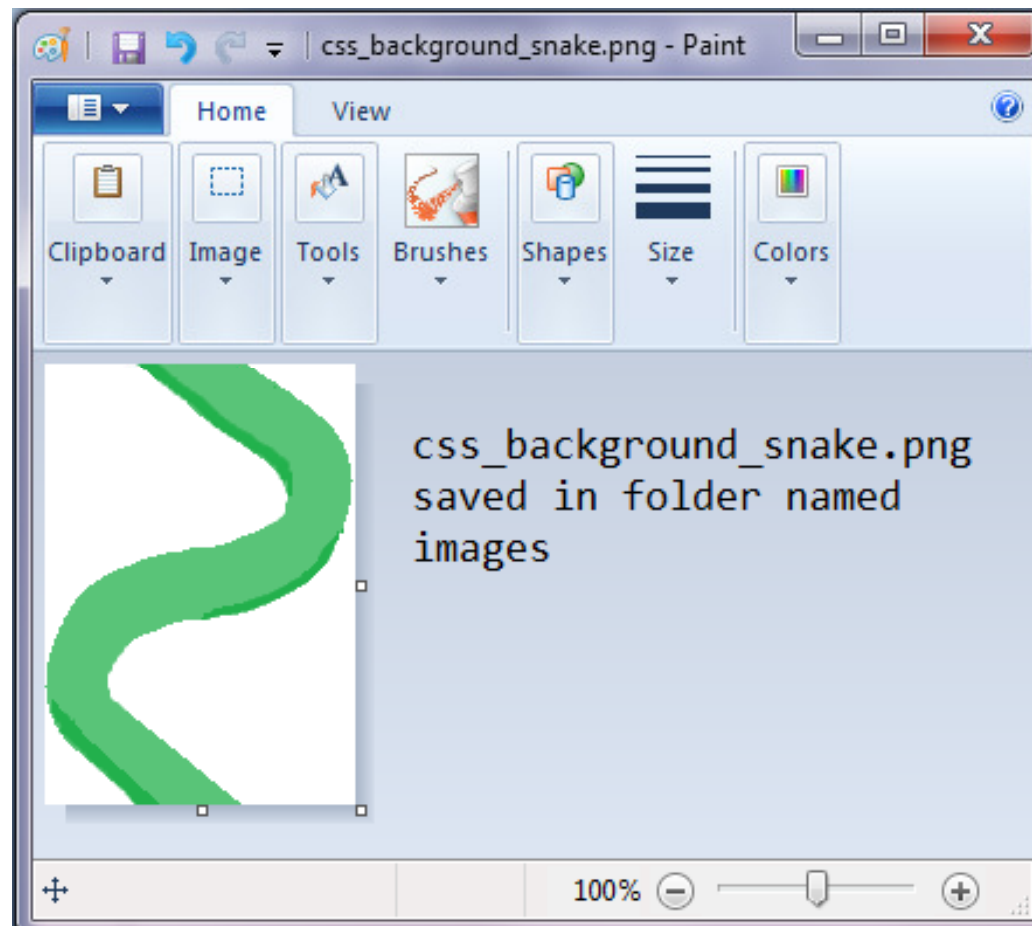
```
li { list-style:  
    url(http://www.example.com/redBullet.png) }
```

# Background Properties

- **color** sets the text colour
- **background-color** can be transparent or a colour value
- **background-image** can be none or a URL value
- **background-attachment** can be scroll or fixed
- **background-repeat** can be no-repeat, repeat, repeat-x, repeat-y
- **background-position** can be a length, top, center, bottom, left, or right



# CSS Background - Image



# CSS Background - Style

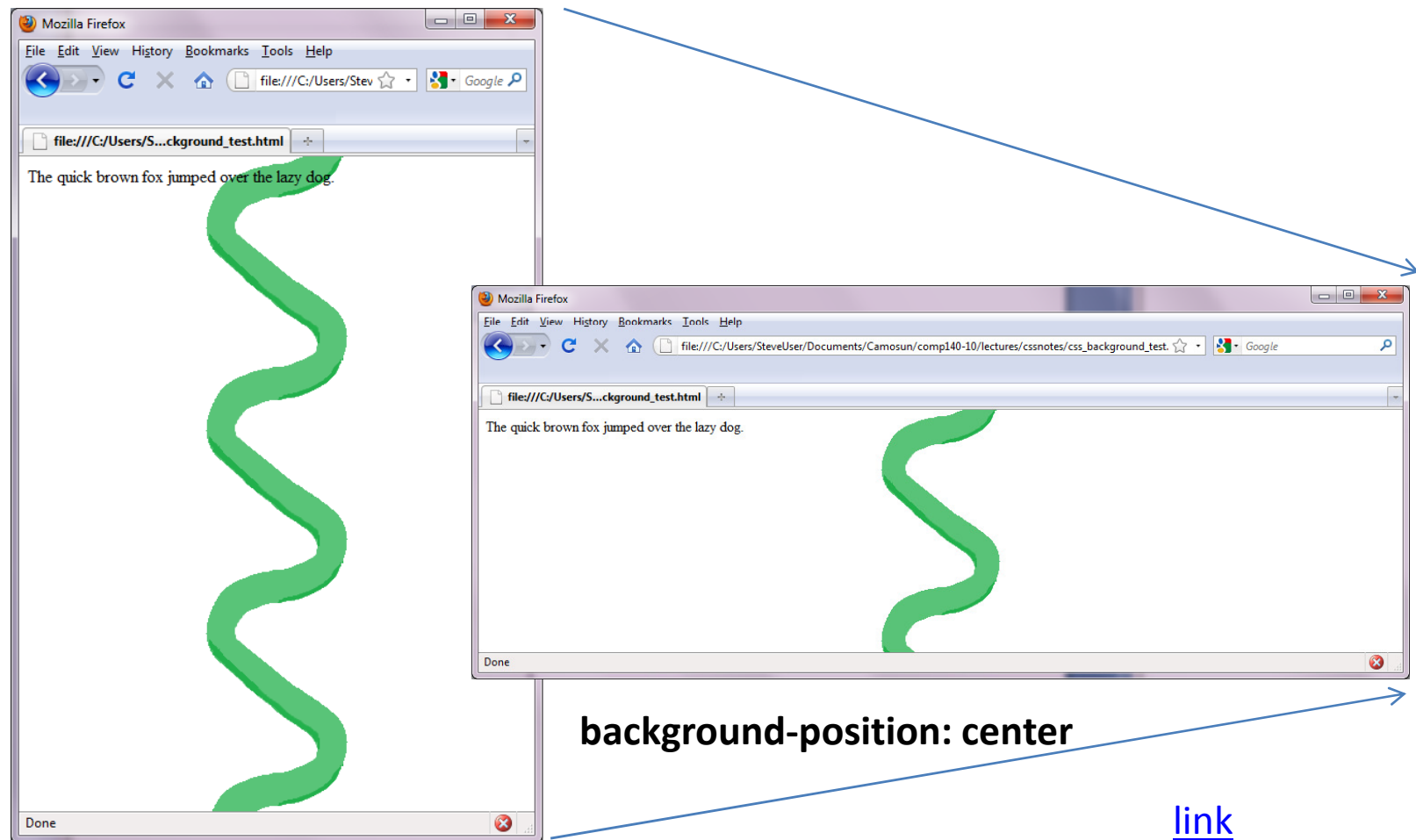
White background with image as the background.

```
1 <html>
2 <style>
3   body { background: white url(../../images/css_background_snake.png);
4         background-repeat: repeat-y;
5         background-position: center;
6       }
7 </style>
8 <body>
9   <p> The quick brown fox jumped over the lazy dog.
10
11   </p>
12 </body>
13 </html>
```

Repeat image along the Y-axis (vertical).

Center the background image within the browser window.

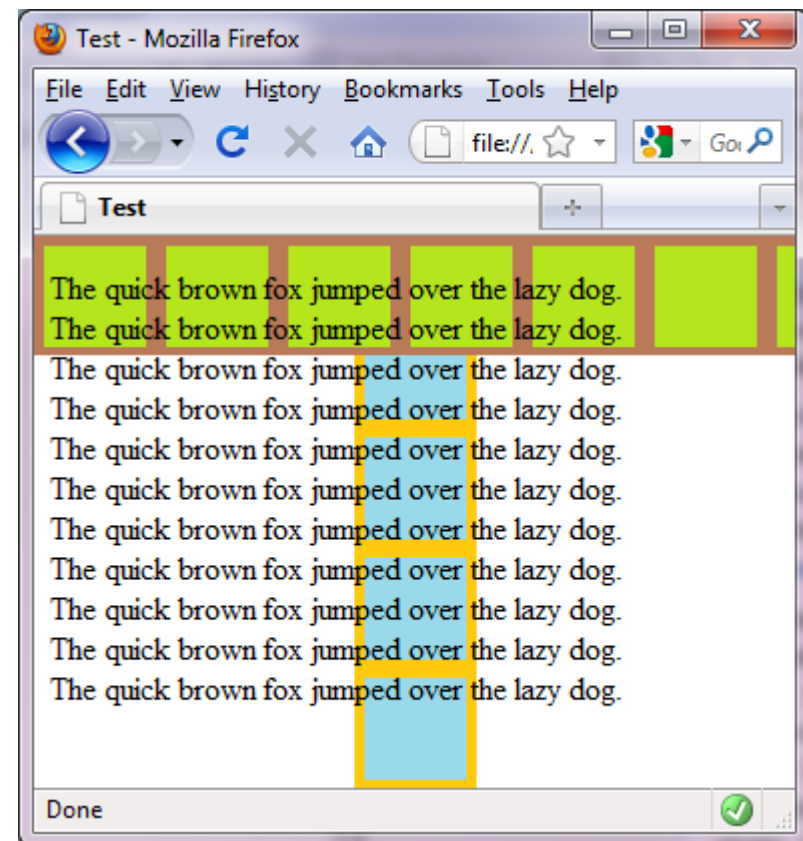
# CSS Background - Resize Window



# CSS Background – repeat x, y

- Does not work on any IE version prior to v9

```
<style type="text/css">
  body { background-image: url(../../images/greenbox.png),
                        url(../../images/bluebox.png);
        background-repeat: repeat-x,
                        repeat-y;
        background-position: top left,
                        center;
  }
</style>
</head>
<body>
<p>
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
The quick brown fox jumped over the lazy dog. <br />
</p>
</body>
</html>
```



# Style Attribute example

- Example of CSS – the **style** attribute is used

```
<h1 style="color:blue">
```

```
This blue header </h1>
```

```
<p style="font-family:Arial;">
```

```
First paragraph </p>
```

```
<h1 style="color:red">
```

```
This red header </h1>
```

```
<p style="font-family:Courier;">
```

```
Second paragraph </p>
```

**This blue header**

First paragraph

**This red header**

Second paragraph

# Style Element example

- Use of the style element in an HTML document

```
<style type="text/css">  
  h1 { color: blue; }  
  h2 { color: red; }  
  p  { font-family: Arial, Helvetica; }  
</style>
```

- Defines style for the tags for this HTML document only
- When two or more separate style elements are defined in the same HTML document, they are all merged together as one element – but when properties conflict – those defined later will override the earlier ones

# CSS Media types

- CSS can support different output depending on the “user agent” (e.g. browser, print, tv, braille, handheld, aural)
- Style sheet definition in the <link> tag
  - <link rel=“stylesheet” type=“text/css” media=“print, handheld” href=“print.css”>
- Or in the style sheet
  - @import url(“loudvoice.css”) aural
  - @media print { ... styles for print }

# CSS Levels of Style


- CSS has three levels of style
  - Inline style
    - Defines the style just for the one occurrence of that element
    - Style attribute is used within the HTML element
  - Embedded style (also called Internal style)
    - Defines a set of tag styles for just the HTML document
    - Style tag is used
  - External style (also called Linked style)
    - Defines a set of tag styles to be used for multiple HTML documents



# Inline Style


- The `style` attribute within the element defines the desired presentation appearance
- CSS `inline` style is used when a specific instance of an element in the HTML requires a unique format

```
<h1 style="color:blue">Blue heading </h1>
```

Style applies only to this text

```
<h1> No style heading</h1>
```

```
<h1 style="color:red">Red heading </h1>
```

Style applies only to this text

# Embedded Style

- Styles for the HTML file's tags are defined within the HTML file inside the <head> section
- The styles are for that HTML document only
- The CSS styles are enclosed in the tag

```
<style type="text/css">
```

```
    h1 { color: blue; } ←
```

```
</style>
```

Note how  
the style is  
defined.

# External Style

- The CSS styles are defined within a separate file  
e.g. file `site.css` contains:

```
h1    { color: blue; }
```

- A `<link>` element is used in the HTML file to indicate the name of the external CSS file
- The `<link>` element is defined in the HTML file's head section

```
<link rel="stylesheet" type="text/css"  
      media="screen" href="site.css">
```

# CSS

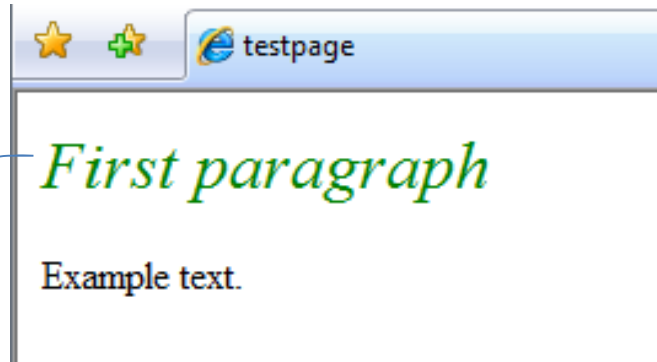
- What this means is that you may define different styles for the same tag

```
1. <html>
2. <head>
3.   <title>testpage</title>
4.   <link rel="stylesheet" type="text/css" href="test.css">
5.   <style type="text/css">
6.     p {color:green;}
7.   </style>
8. </head>
9. <body>
10. <p style="font-size: 20pt;"> First paragraph </p>
11.   Example text.
12. <br>
13. </body>
14. </html>
```

This file  
contains the  
line:

p { font-style: italic; }

The properties are 20pt and  
font style italic and green.



# Cascading

- “Cascading” = Cascading order
  - HTML elements may have different styles applied to them – which one is used by the browser?
  - All styles “cascade” into a single style sheet *in part* by the following **proximity** rule
    - Browser default style -- **lowest** priority
    - External style sheet (a .css file)
    - Embedded style sheet (inside <head> section)
    - Inline style (inside an HTML tag) – **highest** priority
  - there are other, more CSS complicated rules

# Author and User Style sheets

- Users may define their own style sheets
- **Author's** style sheets will override any style defined in the **user's** style sheet
- **User's** style sheet will override the default style sheet
- Internet Explorer: Tools | Options | Accessibility
- Firefox: create the text file `userContent.css` and save it in your Application Data folder under Firefox chrome folder

# CSS Syntax

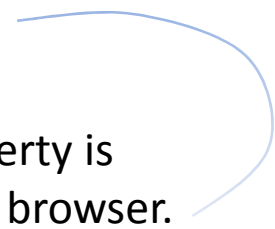
- Always use a **colon** to separate property and value – browsers will not display error messages.
- Selectors, values, properties are case-insensitive – but lowercase encouraged
- Order of properties does **not** matter
- Order of CSS definitions does **not** matter (but if you have duplicate selectors, only the last one is used)
- If the value is multiple words, use double quotes
- Multiple properties can be specified with a semicolon between them
- Do not leave a space between the value number and the units – will not work in Firefox (e.g. “10px” not “10 px”)

# CSS Syntax

- Separate lines for each property for readability
- Comments are enclosed using `/* */`

```
p { color:black;
    font-family: "Times New Roman";
    text-align: left;
    font-size: 15pt; /* test */
    /* font-style: italic; */
}
```

This style property is  
ignored by the browser.





# CSS Selector

- A **selector** is the first part of the CSS style rule and it indicates what HTML is formatted.



# CSS Selector Categories

- Type Selector
- Universal Selector
- Descendant Selector
- Child Selector
- Attribute Selector
- Class Selector
- ID Selector
- Pseudo-classes
- Pseudo-elements

# Type Selector; Universal Selector

- **Type selector** matches the name of an HTML element – every instance of that element in that document

`h1 { font-family: Arial; }` All h1 elements will use this rule.

- **Universal selector** is written as a single **asterisk** and matches any element in the document

`* { font-family: Arial; }` All elements will use this rule.

# Descendant Selectors

- When a style rule needs to be applied to an element contained anywhere within another element, CSS **descendant selectors** are used

**p** **b** { color: red; }

Any **b** elements defined within an **p** element will use this rule.

The **b** element is the **descendant** of **p** element.

**<p>**This is **<b>**important**</b>**  
**</p>**

**<p>** **<div>** This is  
    **<b>**important**</b>** **</div>**

**</p>**

**<h1>**This header is  
    **<b>**not important**</b>**

**</h1>**

These are formatted in red  
-- not this one.

# Child Selectors

- When a style rule is applied to an element's child, a **child selector** is used

**p** > **b** { color: blue; } Any **b** elements defined within an **p** element *as child elements* will use this rule.  
Here the **b** element is the **child** of **p** element.

```
<p>This is <b>important</b>  
</p>  
<p> <div> This is  
    <b>important</b> </div>  
</p>  
<h1>This header is  
    <b>not important</b>  
</h1>
```

This text is formatted in blue  
-- not these ones.

# Attribute Selector

- **Attribute selectors** match when the element sets an attribute in some way

`img[title] { color: blue }`      Matches elements `<img title="text" ... />`

`img[title=start] { color: blue }`      Matches elements `<img title="start" ... />`

`img[title~="blue"] { color: blue }`

Matches elements `<img title="The blue hill" ... />`  
and `<img title="blue rodeo" ... />`

`*[lang|=“en”] { color: blue }`

Vertical bar or “pipe”

Matches any elements having the attribute  
`lang=“en”, or lang=“en-US”, or lang=“en-CA”`

# Class Selector

- In place of an existing HTML tag name, you make up your own name preceded by a period – **class selector**
- Any HTML elements identified by that name as its **class** attribute has that style

```
.headline { font-family: "Courier", serif;  
              color: blue; }
```

```
<b class="headline" >This is bold blue styled text</b>
```


```
<br />
```

```
<p class="headline" >This paragraph is blue too</p>
```

# ID Selector

- Identifies the style for a **unique** instance of the element defined by the ID name – ID selector

```
#menu { text-transform: uppercase; }
```



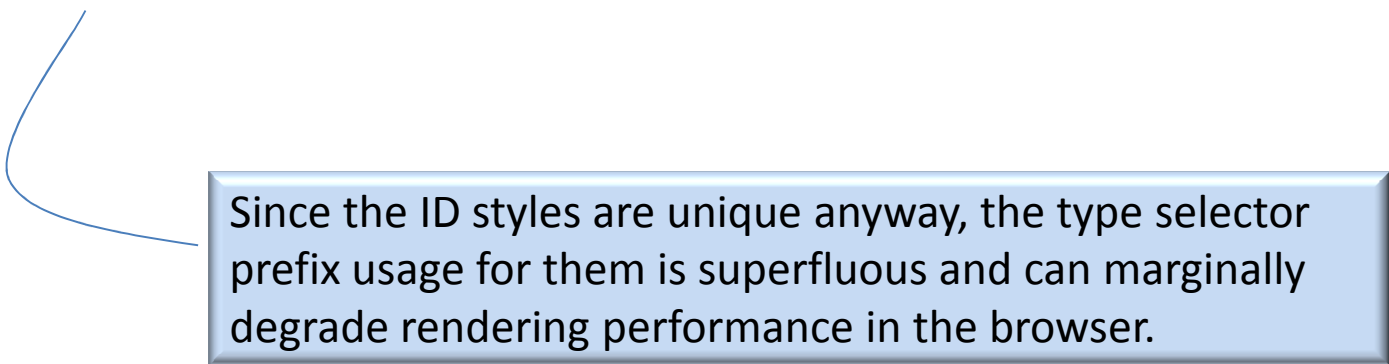
```
<div id="menu"> Text will show as  
uppercase. </div>
```



# Specifying class and ID selectors

- Class and ID style rules can be further specified using a **type selector** as prefix

```
h1.headline    { color: blue;      }  
h2.headline    { color: green;     }  
h3.headline    { color: red;       }  
div#mainpar    { font: Arial;      }  
p#logo         { font-size: 8pt;   }
```



Since the ID styles are unique anyway, the type selector prefix usage for them is superfluous and can marginally degrade rendering performance in the browser.

# Pseudo-classes

- Pseudo-classes are similar to classes except that they refer to CSS specific types of text  
e.g. `:first-child` matches the *first child* of an element

```
div > p:first-child { margin:0; }
```

```
<div class="note">
```

```
<p> Important </p>
```

```
<p> Not important </p>
```

```
</div>
```

This style rule would apply only to this paragraph and not to this one.

# Pseudo-class Links

`a:link` { color: blue; }      a link not yet clicked

`a:visited` { color: red; }      a link you clicked

`a:hover` { color: green; }      as you hover over

`a:active` { color: black; }      as you click link

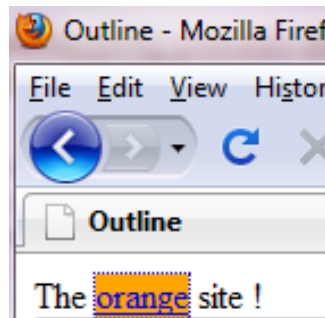
**LVHA** -- definition of CSS style order required

- The style `text-decoration:none` removes the underline on hypertext link text
- [http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS\\_ex03.html](http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS_ex03.html)

# Focus Pseudo-class

- The **:focus** pseudo-class matches any element having **keyboard input** focus (e.g. form input or a link)
- Supported in IE 8, all versions of Firefox, Chrome

```
a:focus { background: orange; }
```



**Focus** occurs only when user **tabs** to this link, not when the cursor is moved there.

# Pseudo-element Selectors

- The :first-letter pseudo-element selects the first *letter (or digit)* of the first line of a text block.

```
p:first-letter { font-size: 3em; float:left;
                font-weight:bold; }
```

<p>Mary had a little lamb...etc</p>

Rendered in the browser as

**M**ary had a  
little lamb,  
little lamb, little  
lamb. Mary had a  
little lamb and

# Group Selectors

- A group of different selectors can share the same style definition – selectors are separated by a **comma**

```
h1, h2, h3    {  color : blue;  }
```

```
p, h1, h2     {  text-align: left; }
```

```
ol, ul        {  margin-right: 20px; }
```

# Classes vs IDs

- Use classes for multiple occurrences of that style in a web page
  - Helpful mnemonic:
    - class has many students
    - ID – the letter I = one
- Use ID when there is only one occurrence of that style in a web page



# Selector Family Tree

- Descendant Selector (the child of the parent)

**div p** (**p** is the child of parent **div**)

- Grandchildren Selector

– Use the universal selector \*

**div \* p** (match all **paragraphs** that are grandchildren or greater of **div** elements)

- Child Selector >

**body > p** (match only children **p** elements of **body**)

- Adjacent Sibling Selector +

**em + p** (match any **p** tags that follow after **em** element)

- Attribute Selector [ ]

**a[title]** (match any **a** tags having a defined **title** attribute)



# Tags **div** and **span**

- The **div** element defines a **block-level** region which may contain text or other content

```
<div style="color:blue;">  
    This is a line of blue text.  
</div>
```

- The **span** element defines an **inline** region

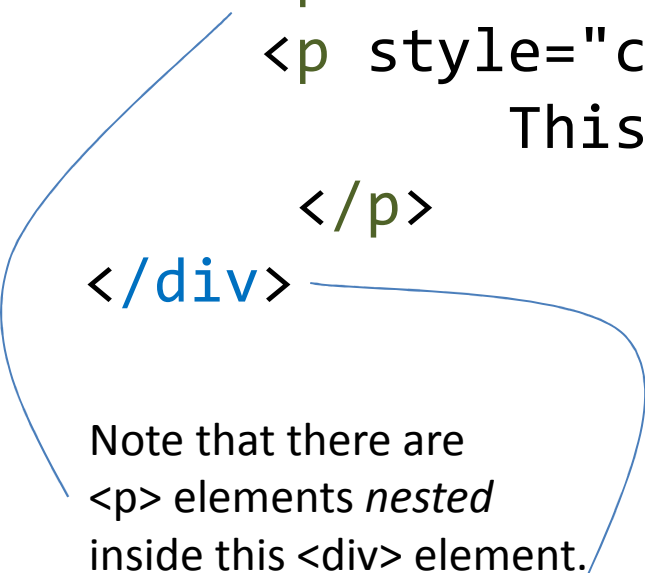
The Delicious apple

is `<span style="color:red;"> red </span>`.

# Div Element

- Nesting elements within a div element

```
<div style="color:blue;">  
  <p>This is the first paragraph.</p>  
  <p style="color:red;">  
    This is the second paragraph.  
  </p>  
</div>
```



Note that there are  
<p> elements *nested*  
inside this <div> element.

This is the first paragraph.

This is the second paragraph.

# CSS Rule of Proximity

- The **closer** style rule will be applied to text
  - **Inline** style **overrides embedded** style
  - In the case of **nested** tags (like `div`), the style rules of the innermost tags override the style rules of the outer tags

```
<div style="color:yellow;"> This is yellow.  
  <div style="color:green;"> This is green.  
    <div style="color:blue;"> This is blue.  
  </div> </div> </div>
```

# CSS Rule of Inheritance

- Many CSS styles are automatically passed into any child elements from the parent element.
  - **font** and **color** styles defined for a parent element will be 'inherited' by any contained child elements by default
  - When inheritance occurs, elements inherit computed values based on the parent and child

```
body { font-size: 10pt }  
h1   { font-size: 120% }
```

```
<body>  
  <h1>Large Heading</h1>  
  ..  
</body>
```

The computed value  
of this text will be  
 $10\text{pt} \times 120\% = 12\text{pt}.$

# CSS Inheritance

```
<style type="text/css">
  .parent { color: red;
            font-family: Arial, sans-serif;
            background-color: wheat;
            margin: 10px;
            padding: 15px;
            border: 1px solid black;
          }
  .child { font-family: Times New Roman, serif;
          }
</style>
</head>
<body style="background-color:yellow;">
<div class="parent">
  The quick brown fox jumped over the lazy dog.
  <div class="child">
    The sixth sheik's sixth sheep's sick.
  </div>
</div>
```

The quick brown fox jumped over the lazy dog.  
The sixth sheik's sixth sheep's sick.

CSS style properties color and font are inherited by the child element.

CSS styles margin, padding and border are not inherited by the child element; otherwise, it would look like this:

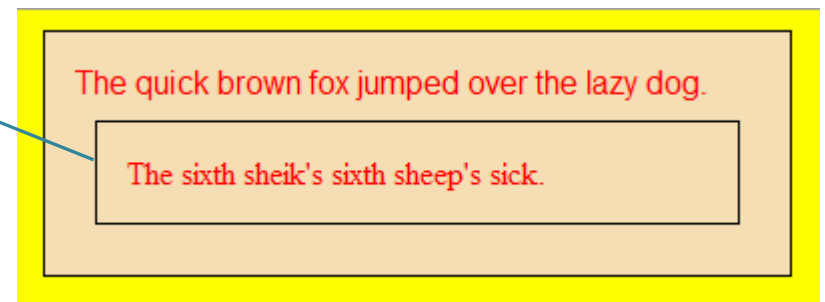
The quick brown fox jumped over the lazy dog.

The sixth sheik's sixth sheep's sick.

# CSS inherit value

- CSS properties not normally inherited can be made to inherit with the 'inherit' value

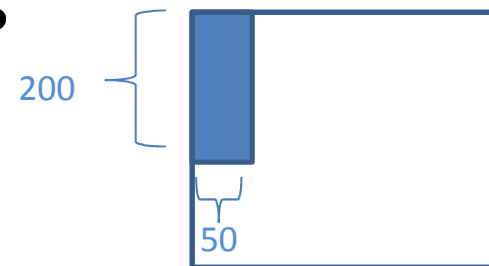
```
<style type="text/css">
  .parent { color: red;
            font-family: Arial, sans-serif;
            background-color: wheat;
            margin: 10px;
            padding: 15px;
            border: 1px solid black;
          }
  .child { font-family: Times New Roman, serif;
           margin: inherit;
           padding: inherit;
           border: inherit;
         }
</style>
</head>
<body style="background-color:yellow;">
<div class="parent">
  The quick brown fox jumped over the lazy dog.
  <div class="child">
    The sixth sheik's sixth sheep's sick.
  </div>
</div>
```



# Image Clipping

- Images can be “clipped” in CSS but only on **absolutely** positioned elements

```
img { position: absolute;  
      clip: rect(0 50 200 0); }
```



Clip the image start at top edge to 50 pixels to right, 200 pixels to the bottom and 0 to the left

<http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/cssClipExample.html>

# CSS Rules

- With these rules CSS needs a formula to determine which styles to apply if a selector has a class style, an ID style and an HTML style

```
<style type="text/css">
    #info { color: blue;      }
    p.note { color: red;      }
    p     { color: green;     }
</style>
</head>

<body>
<p id="info" class="note">
    This will appear blue.
</p>
</body>
```



# Style Priority

- When multiple styles can apply to content, the **ID style overrides a class style**, and the **class style overrides an HTML style**
- Style defined within the tag's style attribute overrides all others.
- This is relevant only when the same properties are defined; otherwise, the properties are all applied

# CSS Rule of Specificity

- CSS has rules to determine which selector's styles will be applied to an element
- Each **HTML** element counts as 1, each **class** style counts as 10, **ID** styles count as 100 – highest total wins

```
<style = "text/css">
```

```
div b { color: blue; }
```

```
b { color: red; }
```

```
</style> </head>
```

```
<body>
```

```
<div>
```

```
<b> This will appear blue. </b>
```

```
</div>
```

← sum is two (div =1, p =1)

← sum is one (p =1)

Because 2 > 1, this text is displayed in blue not red.

# !important rules

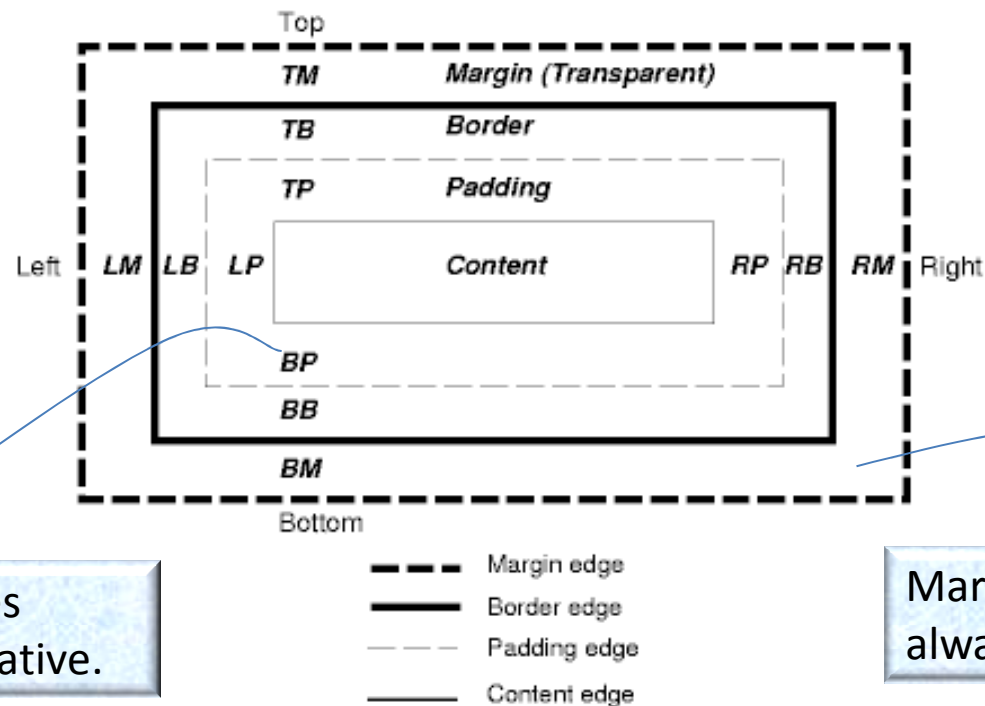
- You can override any style rule with the **!important** declaration

```
p { text-indent: 1em !important;  
    font-style: italic !important;  
    font-size: 16pt !important }
```

If these style rules are defined in the **user's** style sheet, they will override similar style rules in the **author's** style sheet – even if the author used **!important** as well.

# CSS Box properties

- CSS box model defines properties such as margin, padding, border, background, position
- [http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS\\_ex06.html](http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS_ex06.html)

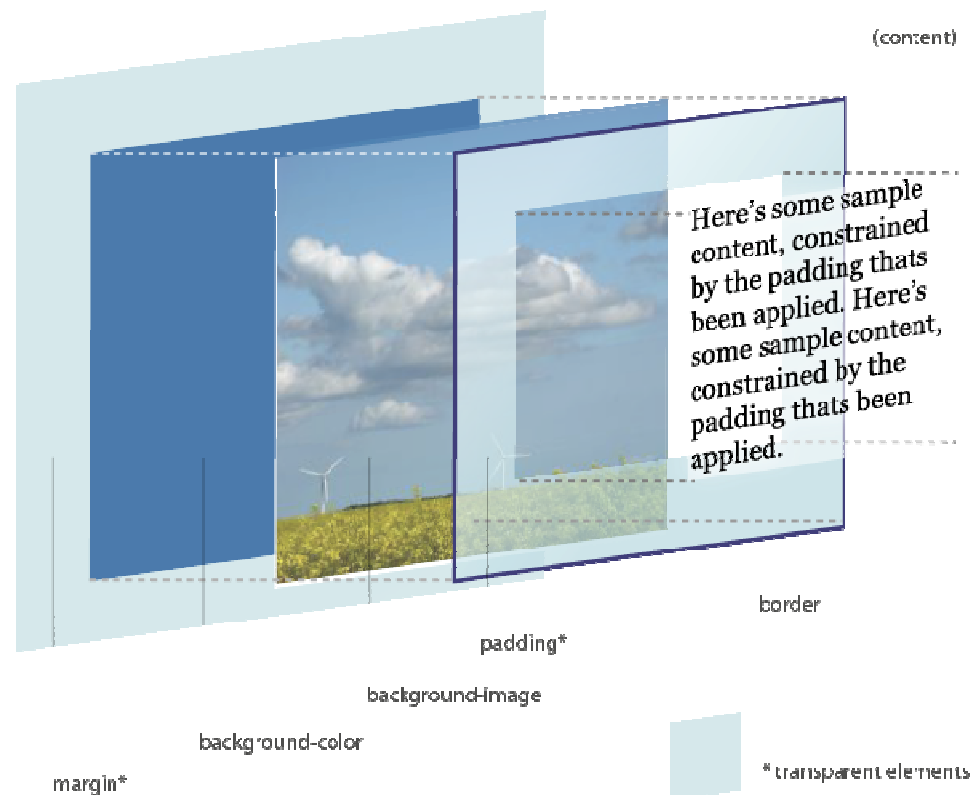


Padding measures must be non-negative.

Margins are always transparent.

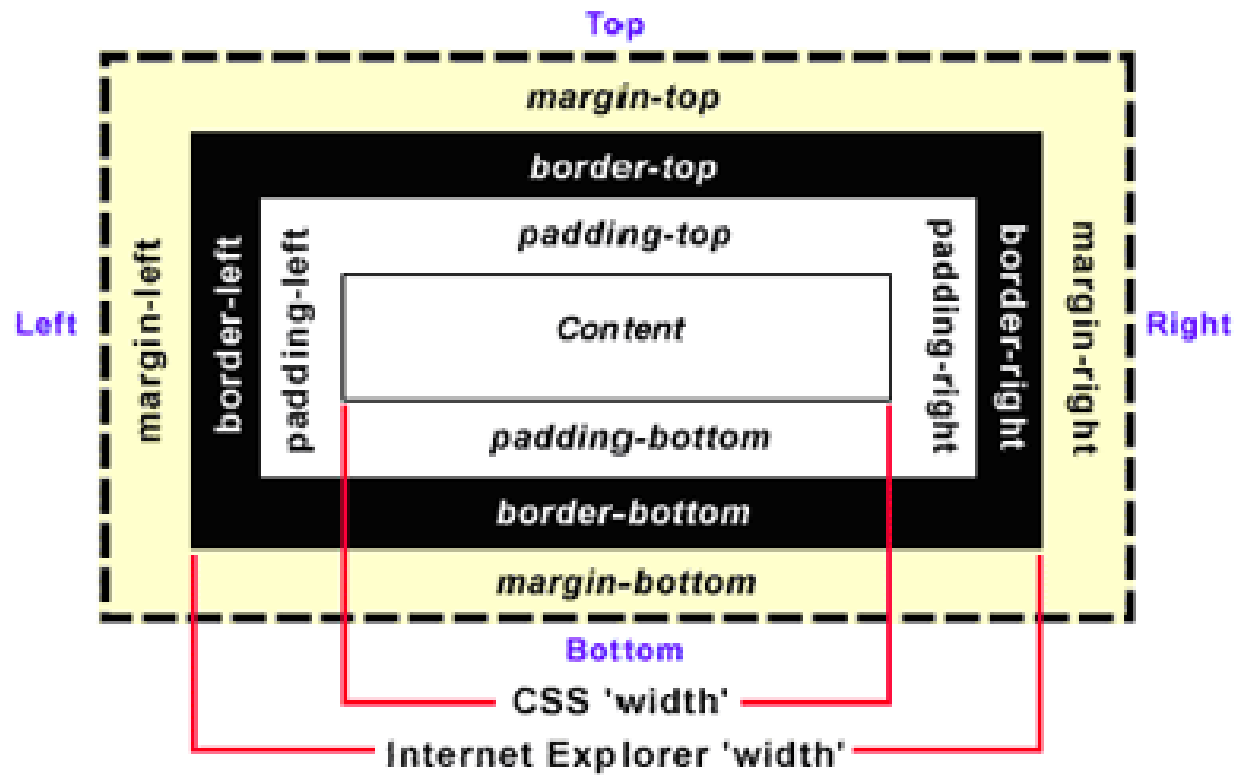
# CSS Box model in 3D

- Box model example



# IE box model – quirks mode

- In **quirks mode** IE browsers understand the box model **width** property their own way:



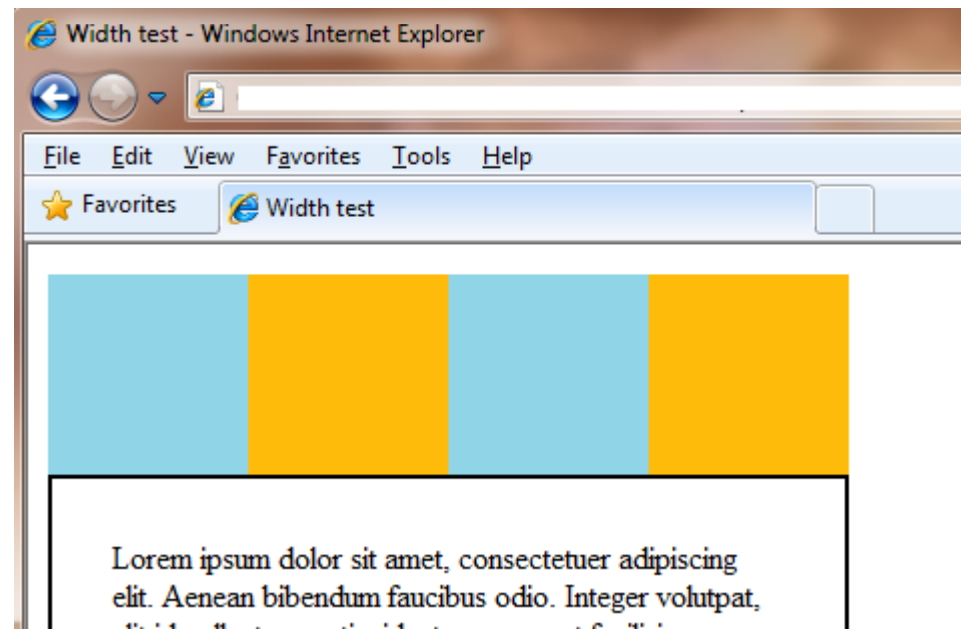
# IE Quirks Mode

- Prior to version 6, IE browsers used a different method for determining the width of an element's box than the method used by the W3C CSS
- Many web sites had already used the non-standard Microsoft implementation of width
- **Quirks mode** refers to a technique used by a browser to make an earlier, non-standard W3C web page compatible for viewing
- Any later version of IE can be “**flipped**” into quirks mode if the HTML has a missing Document Type Declaration (the DOCTYPE)

# Quirks Mode example

- In IE an incorrect or missing DOCTYPE triggers quirks mode

```
<html>
<!-- Missing a fully qualified DOCTYPE so this HTML
      will be rendered in quirks mode in IE browsers.
-->
<head>
<title>Width test</title>
<style type="text/css">
#w1 {
    width:400px;
    padding:30px;
    margin:0px;
    border:solid black thin;
    float:left;
}
```

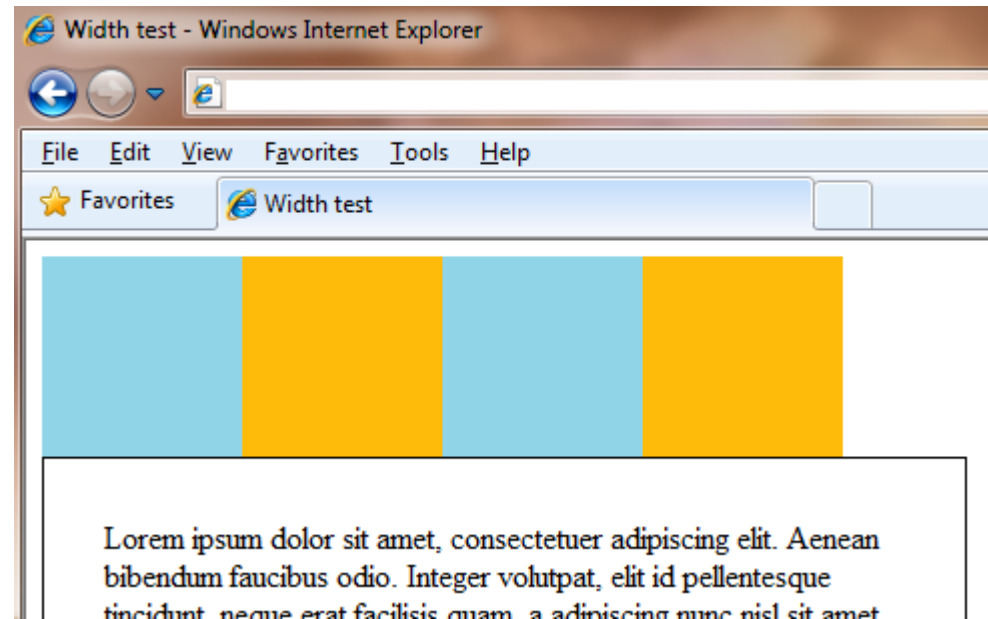




# Standards Mode example

- IE v8 uses standard box model width if a valid DOCTYPE is defined in the HTML

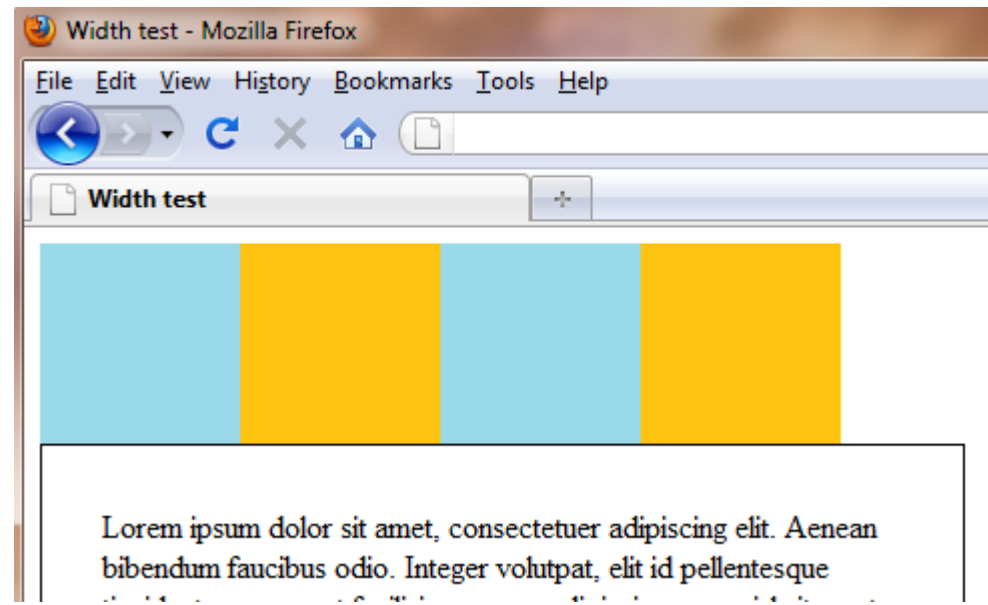
```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<head>
<title>Width test</title>
<style type="text/css">
#w1 {
  width:400px;
  padding:30px;
  margin:0px;
  border:solid black thin;
  float:left;
}
```



# Firefox v3 example

- Firefox uses standard box model width whether the DOCTYPE is valid or not

```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<head>
<title>Width test</title>
<style type="text/css">
#w1 {
  width:400px;
  padding:30px;
  margin:0px;
  border:solid black thin;
  float:left;
}
```



# Positional presentation

- By default CSS uses **normal flow** to place elements on the page
- Or, elements can be placed relative to other elements (**relative positioning**)
- Or, elements can be allowed to **float** to the edge of the page
- Or, elements can be positioned on the page with (x, y) location (**absolute positioning**)

# Normal flow

- Any element not specifying `position:absolute` or `position:fixed` and not `float`
- Block boxes flow vertically starting at the top of their containing block. Inline boxes flow horizontally from left to right.
- The vertical margins of adjoining block boxes are collapsed.

# Relative positioning

- The browser first lays out the element as in normal flow, then the element is displaced by the amount specified by the left or right or top or bottom properties.

```
#content { border: 1px solid;  
            position: relative;  
            left: 100px;  
            top: 120px;  
        }
```

Shifting the content box to the right by 100 pixels and down by 120 pixels.

# CSS Position property - relative

- Relative positioning shifts the referenced item

```
<style type="text/css">
```

```
  p.moveright {position: relative; right: -5em;}
```

```
  p.moveleft  {position: relative; left: -5em;}
```

```
</style>
```

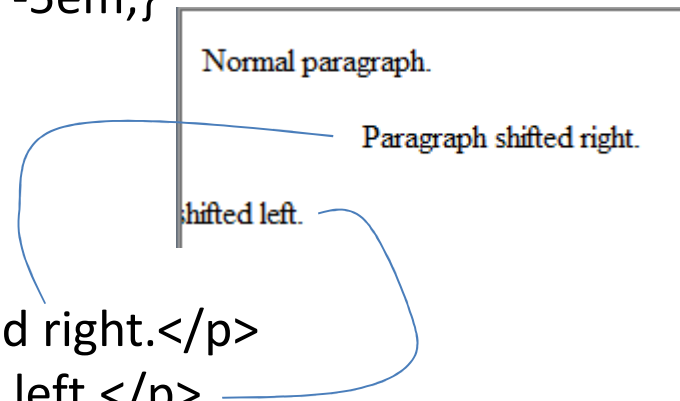
```
</head>
```

```
<body>
```

```
  <p> Normal paragraph.</p>
```

```
  <p class="moveright">Paragraph shifted right.</p>
```

```
  <p class="moveleft">Paragraph shifted left.</p>
```



# Fixed positioning

- Like absolute positioning but the block does not move when a web page is scrolled as other elements do

```
#content { position:fixed;  
           }
```

- Useful for keeping key navigation controls present on screen at all times

# CSS Position property - fixed

- For items that need to stay at specified coordinates regardless of scrolling (IE7 strict mode and Firefox only)

```
<style type="text/css">
    #stay { position: fixed; left:5px; right:5px; }
</style>
</head>
<body>
    <div id="stay"> Fixed in window.</div>
```



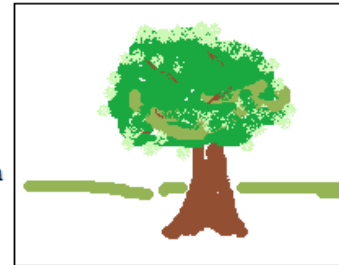
# Float property

- **Float** property is usually applied to an **image** to make it appear to the left or right of surrounding text. You may float **block level** elements only – not inline elements.

`` This is a picture of a tree. ...

Recall block level elements are images, paragraphs, divisions, and lists. Inline elements include spans and line breaks.

This is a picture of a tree. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla



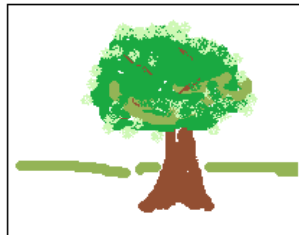
# Float property

- Adding in **margin** property

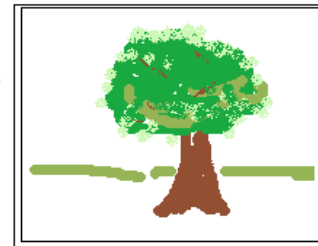
top=0 and right=0  
bottom and left=10 pixels

```
 This is a picture of a tree. ...
```

This is a picture of a tree. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.



This is a picture of a tree. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi.



- Adding in **padding** property and **border**  

```
 This is a picture of a tree. ...
```

# CSS Position property - absolute

- Position property directs where the item is to appear on the display
- Example using absolute positioning:

```
<style type="text/css">
```

```
    #layer1 {position: absolute; left:100; top:100; z-index:0;}
```

```
    #layer2 {position: absolute; left:200; top:200; z-index:1;}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div id="layer1">
```

```
    This is layer 1 in the background<br />positioned at 100,100
```

```
</div>
```

```
<div id="layer2">
```

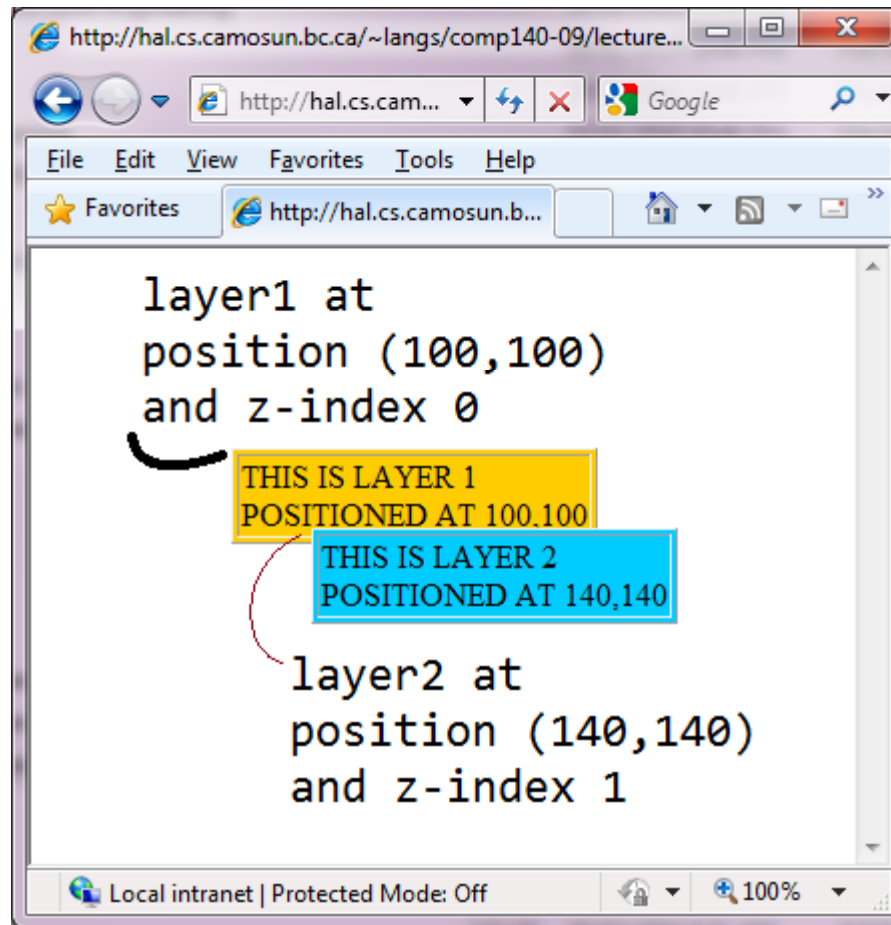
```
    This is layer 2 on top of layer 1<br />positioned at 200,200
```

```
</div>
```

```
http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS\_ex01.html
```

The z-index property indicates the stack level for layered effects. Lower z-index values are on bottom.

# CSS Position Absolute



# Clear property

- CSS property **clear** indicates which sides of an element other **floating elements** are **not** allowed – useful when window is resized

**clear** : right; /\* no floating elements on right \*/

**clear** : left; /\* no floating elements on left \*/

**clear** : both; /\* no floating elements adjacent \*/

**clear** : none; /\* default –

adjacent floating elements allowed \*/

# Display property

- The display property controls the type of box an element generates
- For example you may want to force some **inline** elements into **block** elements

```
a.menu    { display: block; }  
h1        { display: inline; }  
p.newad   { display: inline-block; }  
div.rem   { display: none; }
```

inline-block makes the element generate a block box that is laid out as if it were an inline box. None means no display.

# Lists

- Applies to the list item element for unordered and ordered lists
- `list-style-type` can be disc, circle, square, decimal, lower-roman, upper-roman, lower-alpha, upper-alpha, none
- `list-style-image` can be none or a URL value
- `list-style-position` can be inside or outside
- [http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS\\_ex02.html](http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS_ex02.html)

# List style type

- List style types can be alphabetic, numeric, or symbols

list-style-type: none

- list-style-type: disc

- list-style-type: circle

- list-style-type: square

- 1. list-style-type: decimal

- i. list-style-type: lower-roman

- I. list-style-type: upper-roman

- a. list-style-type: lower-alpha

- A. list-style-type: upper-alpha



# Cursors

- Cursor property specifies the type of cursor to display when pointing on an element
- cursor can be default (arrow), crosshair, pointer, move, url, text, wait, help
- cursor can also be e-resize (resize left-right), n-resize (resize up-down), nw-resize, ne-resize, se-resize, sw-resize
- [http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS\\_Cursors.htm](http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/CSS_Cursors.htm)

# Outlines

- It may be appropriate to provide an **outline** surrounding a visual object like a button or an active form field to make them stand out

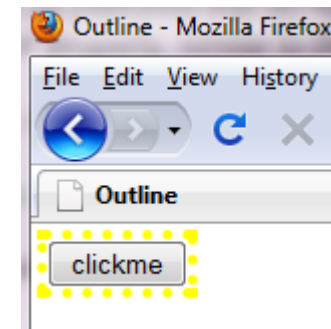
```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<head>
<title>
  Outline
</title>
<style type="text/css">

  input { outline: dotted thick yellow; }

</style>
</head>
<body>

  <form>
    <input type="submit" name="clickme" value="clickme" />
  </form>

</body>
</html>
```



# Outlines

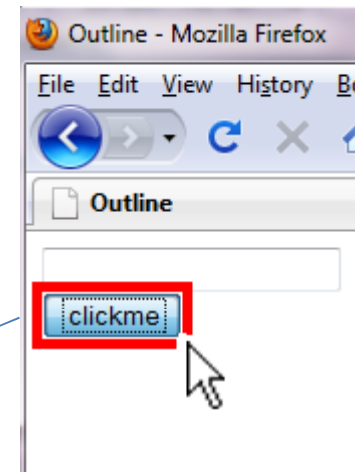
```
<!DOCTYPE html PUBLIC
  "-//W3C//DTD XHTML 1.0 Transitional//EN"
  "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<head>
<title>
  Outline
</title>
<style type="text/css">

  input:focus { outline: dotted thick yellow; }
  input:active { outline: thick solid red; }

</style>
</head>
<body>

  <form>
    <input type="text">
    <br />
    <input type="submit" name="clickme" value="clickme" />
  </form>

</body>
</html>
```



# Overflow

- When text won't fit the size of the given box, what do you want happen?

```
<style type="text/css">
```

```
#a1 { overflow: scroll;
      width: 200px;
      height: 200px;
      float: left;
    }

#a2 { overflow: hidden;
      width: 200px;
      height: 200px;
      margin-left: 30px;
      display: inline-block;
    }

#a3 { overflow: auto;
      width: 200px;
      height: 200px;
      margin-left: 30px;
      display: inline-block;
    }

#a4 { overflow: visible;
      width: 200px;
      height: 200px;
      margin-top: 40px;
    }
```

overflow: scroll

Our booked passenger showed in a moment that it was his name. The guard, the coachman, and the two other passengers eyed him distrustfully. 'Keep where you are,' the guard called to the voice in the mist, 'because, if I should make a mistake, it could never be set right in

overflow: hidden

Our booked passenger showed in a moment that it was his name. The guard, the coachman, and the two other passengers eyed him distrustfully. 'Keep where you are,' the guard called to the voice in the mist, 'because, if I should make a mistake, it could never be set right in your lifetime. Gentleman of the name of Lorry answer straight.' 'What is the

overflow: auto

Our booked passenger showed in a moment that it was his name. The guard, the coachman, and the two other passengers eyed him distrustfully. 'Keep where you are,' the guard called to the voice in the mist, 'because, if I should make a mistake, it could never be set right in your lifetime. Gentleman of the

Hidden means  
any overflow is clipped.

overflow: visible

Visible is the default  
any overflow is not  
clipped.

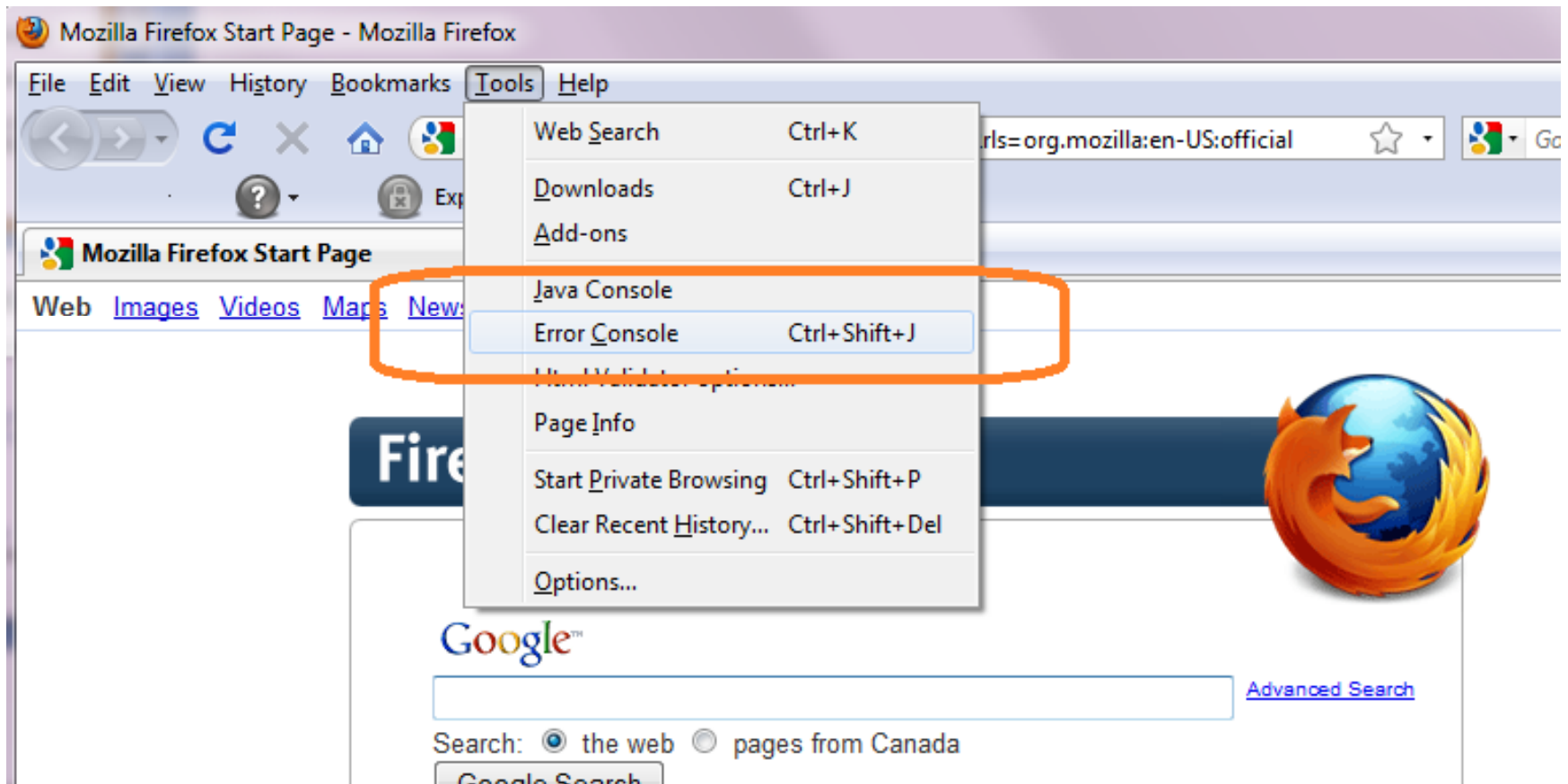
Our booked passenger showed in a moment that it was his name. The guard, the coachman, and the two other passengers eyed him distrustfully. 'Keep where you are,' the guard called to the voice in the mist, 'because, if I should make a mistake, it could never be set right in your lifetime. Gentleman of the name of Lorry answer straight.' 'What is the matter?' asked the passenger,

<http://www.cs.camosun.bc.ca/~langs/comp140-10/lectures/cssnotes/overflow.htm>

# Overflow CSS3

- overflow-x overflow-y
- **Overflow-x** determines clipping at the left and right edges (scrollbar is horizontal)
- **Overflow-y**, at top and bottom edges (scrollbar is vertical)
- CSS3 has defined an **overflow-style** but no browser supports this property yet

# Firefox Error Console



# CSS Errors Example

[link](#)

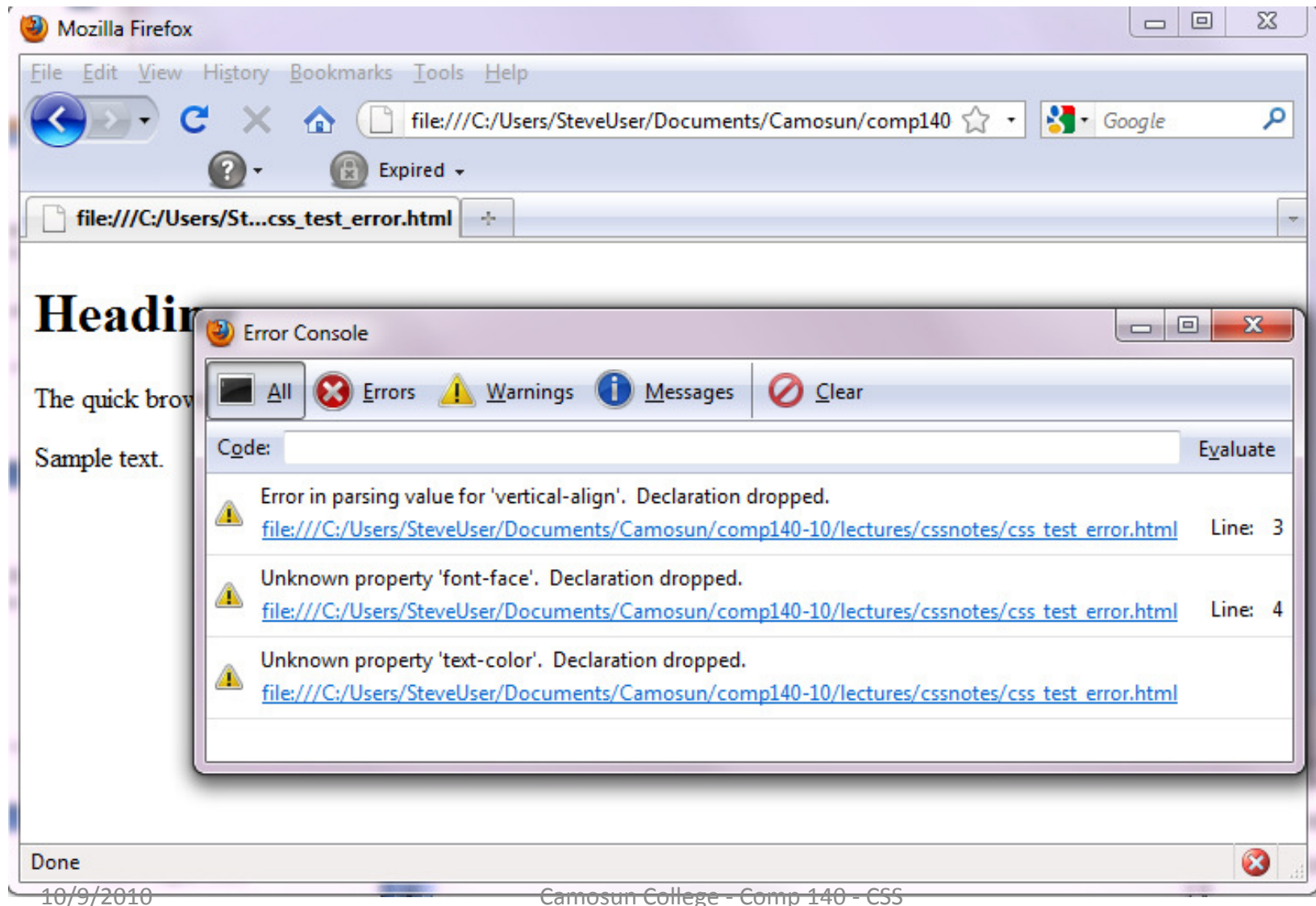
The image shows a Notepad window titled 'css\_test\_error.html - Notepad'. The window contains HTML code with several CSS-related errors highlighted by yellow boxes and annotated with blue callout boxes. The code is as follows:

```
<html>
<style>
  h1 { vertical-align:center;
        font-face:"Times Roman"; }
</style>
<body>
  <link href="abc.css" type="text/css">
  <h1 style="text-color:green;">Heading</h1>
  <p abc:"xyz;">The quick brown fox jumped over the lazy dog.
</p>
  <p style="line-height:1;">Sample text.
</p>
</body>
</html>
```

The errors and their annotations are:

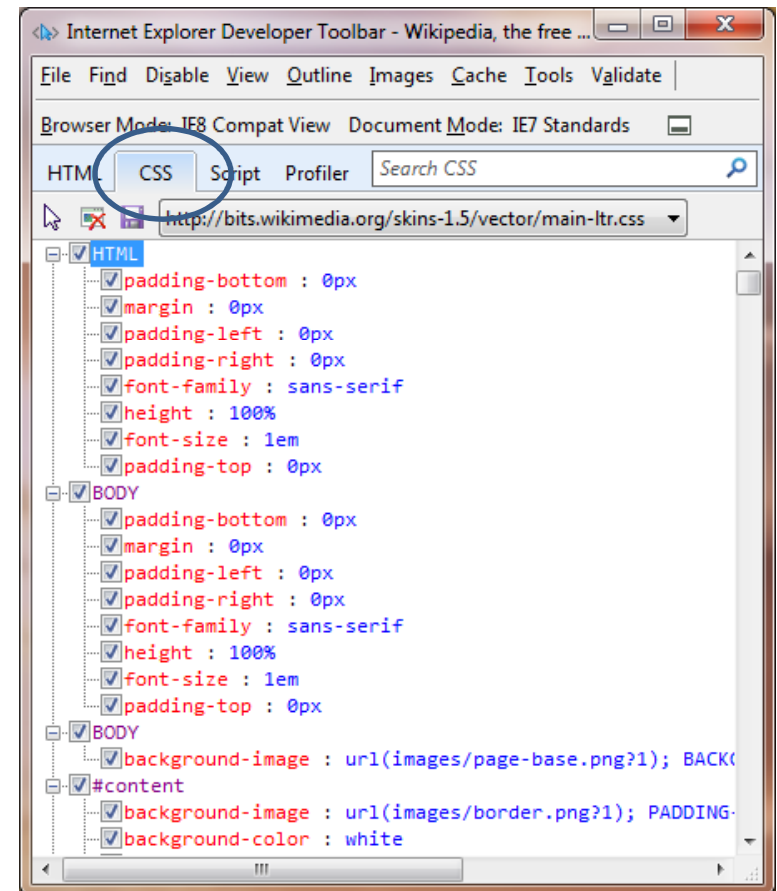
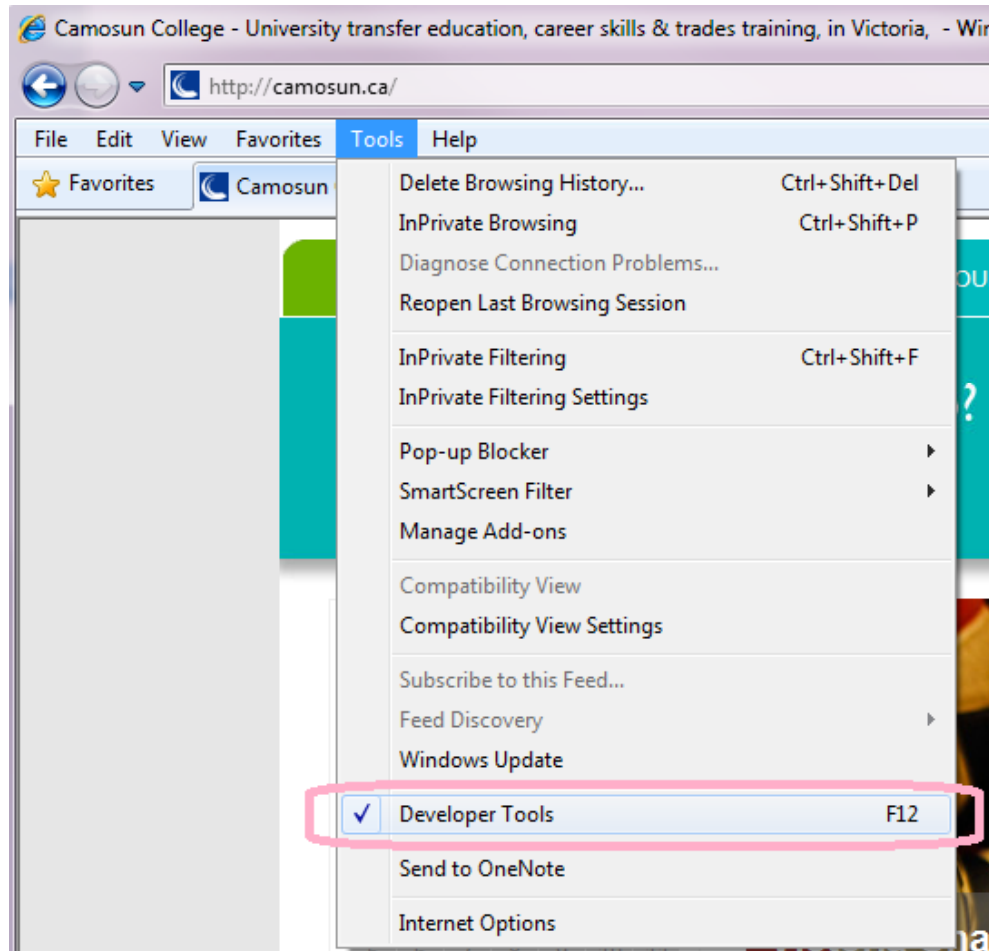
- Incorrect value name, should be middle.** Points to the `vertical-align:center` property in the `h1` style rule.
- Incorrect property name, should be font.** Points to the `font-face` property in the `h1` style rule.
- Incorrect location for link element.** Points to the `<link>` element, which is placed inside the `<body>` tag instead of the `<head>` tag.
- Incorrect property name, should be color. Unknown property name abc** Points to the `abc:"xyz;"` in the `<p>` tag.
- Nonzero CSS values should include the measure (e.g. em or px)** Points to the `line-height:1;` in the `<p>` tag.

# CSS Errors caught by Firefox





# Microsoft Internet Explorer Developer Toolbar



# W3C Unified Validator

The screenshot shows the W3C Unified Validator web interface within a Windows Internet Explorer browser window. The browser's address bar displays the URL `http://validator.w3.org/unicorn/`. The page features a blue header with the W3C logo and the title "Unicorn - W3C's Unified Validator" with the tagline "Improve the quality of the Web". Below the header, there are three tabs: "By URI", "By File Upload", and "By Direct Input". The "By URI" tab is selected, and the "Select a task" section shows a dropdown menu for "General Conformance Check" with the description "Performs as many checks as possible." Below this, the "Validate by URI" section prompts the user to "Enter the URI of a document you would like checked" and includes a text input field labeled "Address:" and a "Check" button. A green-bordered box contains a message about community support with a "Donate" link. At the bottom, there are links for "Documentation", "Download", "Feedback", and "Translations", along with a copyright notice and an "open source" logo.

Checks HTML,  
CSS, and RSS  
for compliance  
to the standard  
grammar rules.

# Review CSS Questions

# References

- <http://www.w3.org/TR/CSS21/cover.html>
- <http://www.devwebpro.com/9-top-css-essential-skills-that-every-web-designer-should-learn/>
- [http://dba.med.sc.edu/price/irf/Adobe\\_tg/models/hsb.html](http://dba.med.sc.edu/price/irf/Adobe_tg/models/hsb.html) (HLS information)
- <http://www.impressivewebs.com/css-opacity-reference/>
- <http://www.hicksdesign.co.uk/boxmodel/>