

# Webpage Markup with HTML5

(C) Pravin Pawar - SUNY Korea, Paul Wang - Kent State University, Ohio

# USEFUL RESOURCES

- [https://www.w3schools.com/html/html5\\_intro.asp](https://www.w3schools.com/html/html5_intro.asp)
- <https://developer.mozilla.org/en-US/docs/Web/HTML>
- <http://codepen.io/pen/>

# HTML5 PAGE STRUCTURE

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml"
      lang="en" xml:lang="en">
<head>
<meta charset="utf-8"/>
<title>Great Company: homepage</title>
<!-- other head elements as appropriate -->
</head>
<body>  <!-- page content begin -->
        . . .
        . . .
<!-- page content end -->  </body>
</html>
```

XML namespaces help contextualize elements and attributes, among other things. It also offers a precise identification for a particular element or attribute.

For instance, the `<html>` element can be defined by anyone and have any meaning. However, the `<html>` element within the `http://www.w3.org/1999/xhtml` namespace is unique and refers to the XHTML.

# PAGE TITLE

1. Displayed in the *title bar* of the browser window
2. Used in making a bookmark for the page

# CREATING A WEBPAGE

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en"
      xml:lang="en">
<head>
<meta charset="utf-8"/>
<title>My Sample Webpage</title>
</head>
<body style="background-color: cyan; margin: 50px">
<h2>Hi everybody!</h2>
<p>My Name is (put your name here) and today is
<time>(put in the date yyyy-mm-dd)</time>.</p>
<p>HTML5 is cool.</p></body></html>
```

Demo: **Ex:** FirstPage

# SAMPLE WEBPAGE



# HTML5 ELEMENTS

- There are more than 100 different elements in HTML5.
- Elements for meta information are placed in the `head` element.
- Elements for page content are placed in the `<body>` element.
- HTML5 distinguishes between two broad types of content elements:
  1. *flow elements* that occupy their own vertical space in a page and *phrasing elements* that act like words and phrases.
  2. Flow elements act like paragraphs, lists, and tables and can contain other flow elements, phrasing elements, and texts.
  3. Phrasing elements can contain other phrasing elements and texts.

# HTML5 ELEMENTS CLASSIFIED

- *Top-level elements:* `html`, `head`, and `body`.
- *Head elements:* elements placed inside `head`, including `title` (page title), `style` (rendering style), `link` (related documents), `meta` (data about the document), `base` (URL of the document), and `script` (client-side scripting). These elements are not part of the page display.
- *Block-level elements:*
  - Flow elements behaving like paragraphs, including `article`, `h1`–`h6` (headings), `header`, `footer`, `section`, `p` (paragraph), `figure`, `canvas` (dynamic drawing area), `pre` (preformatted text), `div` (designated block), `ul`, `ol`, `dl` (lists), `table` (tabulation), `form` (user-input forms), and `video` (video).
  - A block element occupies 100% of the available width to it and will be *stacked vertically* with preceding and subsequent block elements.
  - A block-level (or simply block) element always starts on a new line, and any element immediately after the block element also begins on a new line.



- *Inline elements:*

- Phrasing elements behaving like words, characters, or phrases that flow horizontally to fill the available width before starting new lines.
- Usually, inline elements are placed within block elements.
- Inline elements include `a` (anchor or link), `audio` (sound), `br` (line break), `code` (computer code), `img` (picture or graphics), `em` (emphasis), `nav` (navigation), `samp` (sample output), `span` (designated inline scope), `strong` (strong emphasis), `sub` (subscript), `sup` (superscript), `time` (time/date), and `var` (variable name).

# HTML5 ENTITIES

- In an HTML5 document, certain characters, such as `<` and `&`, are used for markup and must be *escaped* to appear literally. HTML provides *entities* (*escape sequences*) to introduce such characters into a webpage. For example, the entity `&lt;` gives `<` and `&amp;` gives `&`.
- Characters not on the regular keyboard can also be included directly or using HTML5-defined character references.
- Use `<html>` `<head>` `<meta charset="utf-8">`
- E.g. displaying Korean characters:
- <https://ofcourse.kr/html-course/%EC%9D%B8%EC%BD%94%EB%94%A9>

# EVOLUTION OF HTML

- In 1989, Tim Berners-Lee defined a very simple version of HTML based on SGML. The first common standard for HTML was HTML 3.2 (1997).
- HTML 4.01 became a W3C (the World-Wide Web Consortium) recommendation in December 1999.
- HTML 4.01 begins to clearly separate the document structure and document presentation and specifies a clear relationship between HTML and client-side scripting (JavaScript).
- In January 2000, W3C released XHTML 1.0 as an XML reformulation of HTML 4.01.
- XHTML pages can be processed easily by any XML tool.

- HTML5 combines the XHTML, HTML 4, and CSS3 standards, introduces new elements and APIs, as well as incorporates MathML (Mathematical Markup Language) and SVG (Scalable Vector Graphics) into HTML.
- HTML5 can also easily be written in an XML compliant way. The release of the HTML5 standard promises to bring significant advantages to Web developers and benefits to end users.
  - SVG example:  
[https://www.w3schools.com/graphics/svg\\_examples.asp](https://www.w3schools.com/graphics/svg_examples.asp)
  - MathML example: <https://www.w3.org/Math/XSL/csmall2.xml>  
(works with firefox)

# HTML5 INTEGRATION



# WEBPAGE SYNTAX

- All tags begin with `<` and end with `>`. The *tag name* is given immediately following the leading `<`. Make sure the tag is spelled correctly. Unrecognized tags are ignored by browsers. Any *attributes* are given following the tag name in the form:

`<tag attribute1="value" attribute2="value" ... >`

You may use single quotes (`'`), instead of double quotes (`"`), for the value part of any attribute. Be careful; forgetting to close a quote can result in a blank page display.

e.g. `<a href="https://www.w3schools.com">This is a link</a>`

- Tag names and attributes are lowercase. Attributes are always given in either of the two forms:

`attribute_name="value"`

`attribute_name='value'`

For *Boolean attributes*, those that are either on or off, use either of these forms for “on”:

`attribute_name="attribute_name"`

`attribute_name=""`

and omit the attribute for “off”.

- Unrecognized tags and attributes are ignored by browsers.
- Most elements involve start and end tags. Other elements, such as `<br/>` (line break) and `<img .../>` (image), do not have closing tags and are known as *void elements*.
- Elements must be *well-formed*. This means no missing opening or closing tags and no improper element nesting. For example, `<p>Learning <strong>HTML5</p></strong>` overlaps the tags and is not properly nested. The correct nesting is

`<p>Learning <strong>HTML5</strong></p>`

- Attributes can be required or optional and can be given in any order. If an attribute is not given, its initial (default) value, if any, is used.
- Extra white space and line breaks are allowed between the tag name and attributes and around the = sign inside an attribute. Line breaks and white space within attribute values are also allowed but should be avoided because they may be treated inconsistently by browsers.



# HTML5 CORE ATTRIBUTES

- `id`—Uniquely identifies the element in a page. All `ids` in a document must be distinct. Among other uses, a URL ending in `#some_id` can lead directly to an element inside a document.
- `style`—Gives presentation styles for the individual element.

For example, the code

```
<body style="background-color: cyan">
```

gives the color value `cyan` to the style property

`background-color` for this element. Several style properties separated by semicolons can be given. The `style` attribute is a direct but inflexible way to specify presentation style. (css is used for improved styling)

- `class`— Specifies a *style class* or a space separated list of style classes for the element. For example, `class="fineprint"` or `class="footnote fineprint"`. Thus, you may place HTML elements in different classes and associate presentation styles to all elements belonging to the same class.
- `title`—Provides a title for the element. This may be used for tool-tip displays by browsers.
- `hidden`—Prevents the element from being displayed by a browser when set to `true`.

# WEBPAGE ARCHITECTURE

- A typical webpage is organized into the following parts inside the root element `html`.
- The `head` element contains child elements: the page title (`title`), the page character encoding with a `meta` tag `<meta charset="UTF-8"/>` both are required by HTML5. Usually `head` contains additional elements for styling, scripting, and other meta info.
- The `body` element provides the page content, often organized into a `header` part for the top banner and a horizontal navigation bar at the top of the page.
- After the header, the page may also have a vertical navigation bar on the left side. The flow (block) element `nav` is used for navbars that organize links.

# NAV EXAMPLE

```
<nav class="crumbs">
```

```
  <ol>
```

```
    <li class="crumb"><a href="bikes">Bikes</a></li>
```

```
    <li class="crumb"><a href="bikes/bmx">BMX</a></li>
```

```
    <li class="crumb">Jump Bike 3000</li>
```

```
  </ol>
```

```
</nav>
```

```
<h1>Jump Bike 3000</h1>
```

```
<p>This BMX bike is a solid step into the pro world. It looks as legit as it rides  
and is built to polish your skills.</p>
```

- Then, there are one or more articles (`article`) for the main content, followed by a `footer` at the end of the webpage.
- An article may contain one or more sections (`section`) that consist of headings (`h1` through `h6`), paragraphs (`p`), tables, figures, audio, and video.
- A paragraph may contain text, pictures (`img`), audio, and video.
- The footer often provides information on copyright, author, and links to terms of use, privacy policy, customer service, and so on.
- An `aside` flow (block) element can set aside related information, such as links to references, outside resources, and advertisements, that are not the primary content of the page.

# ARTICLE EXAMPLE

```
<!DOCTYPE html>
<html>
<body style="background-color:powderblue;">

<article class="film_review">
  <header>
    <h2>Jurassic Park</h2>
  </header>
  <section class="main_review">
    <p>Dinos were great!</p>
  </section>
  <section class="user_reviews">
    <article class="user_review">
      <p>Way too scary for me.</p>
      <footer>
        <p>
          Posted on
          <time datetime="2015-05-16 19:00">May 16</time>
          by Lisa.
        </p>
      </footer>
    </article>
  </section>
</body>
</html>
```

# SECTIONS AND PARAGRAPHS



```
<article>
<h1>The Green Earth Project</h1>
<section>
<h3>Project Background</h3> <!-- section 1-->
  <p>Put first paragraph here</p>
  <p>Put second paragraph here</p>
<h4>A Successful Past</h4><!-- subsection -->
  <p>Another paragraph here</p>
</section><section>
<h3>Current Status of Green Earth</h3><!-- section 2-->
  <p>Another paragraph here</p>
</section><section>
<h3>Future Goals</h3><!-- section 3-->
</section></article>
```

Demo: **Ex:** GreenEarth



# FLOWING CONTENT

- The `p` (a paragraph) is a flow (block) element which may contain texts and phrasing elements.
- A flow element is typically displayed with a leading and a trailing blank line.

The element content will be formatted to fit the available width. Line breaks are inserted automatically (*line wrapping*) where needed to render the contents. Extraneous white spaces between words and lines within the source text of the content are normally ignored (*white-space collapsing*).

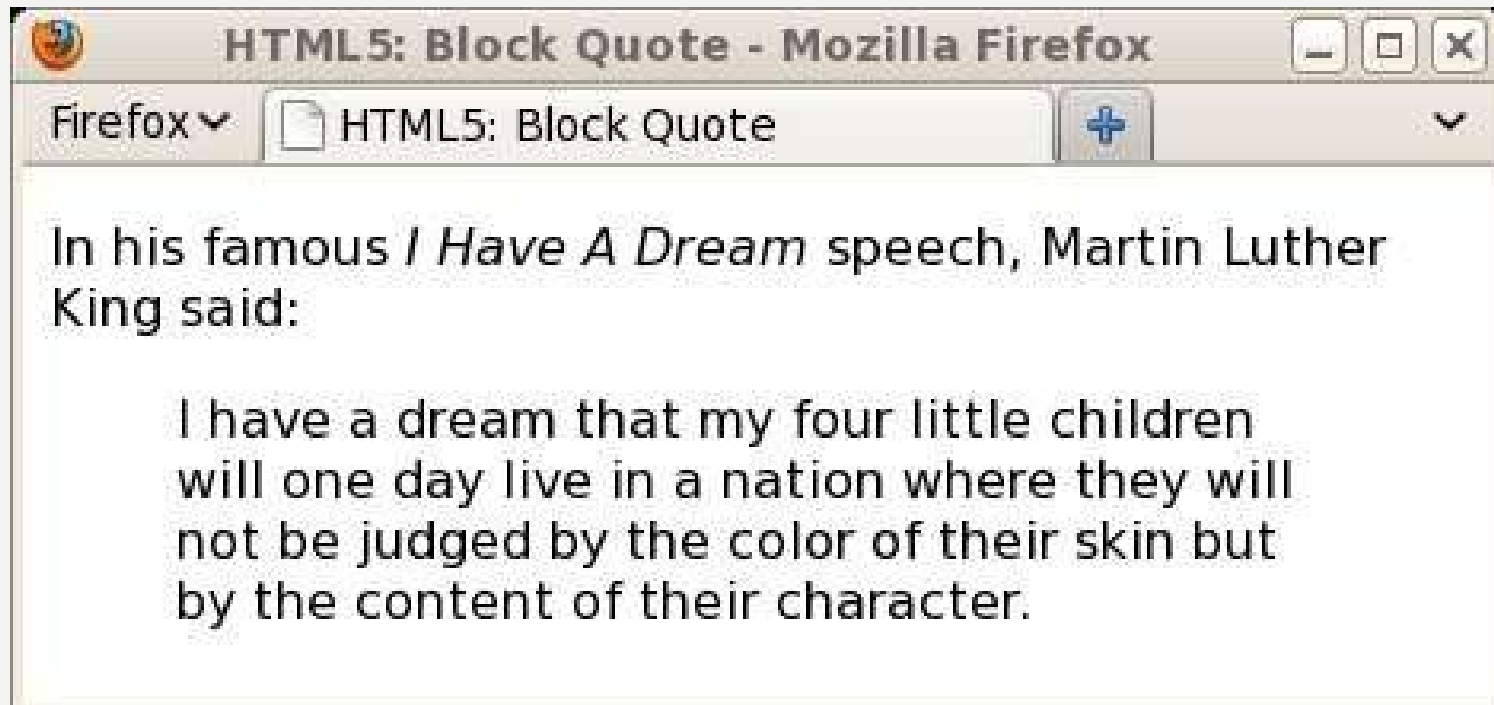
- If you need a line break at a specific point in the content, you can use the `<br />` tag to call for a line break. For a long-running text without spaces, such as an email or Web address, you can insert the void element `<wbr />` to indicate a

*line break opportunity*. The browser will do a line break indicated by `wbr` only if necessary. For example,

```
<p>Please visit  
www.somelong.<wbr />andcomplicated.com.</p>
```

- Inside a flow element, you can place other phrasing (inline) elements such as `q`, `em`, `mark`, `strong`, `img`, `video`, and `audio`.

# BLOCK QUOTE



<p>In his famous <em>I Have A Dream</em> speech,  
Martin Luther King said:</p>

<blockquote cite="<http://www.mlkonline.net>">

I have a dream that my four little children will one  
day live in a nation where they will not be judged  
by the color of their skin but by the content of  
their character.

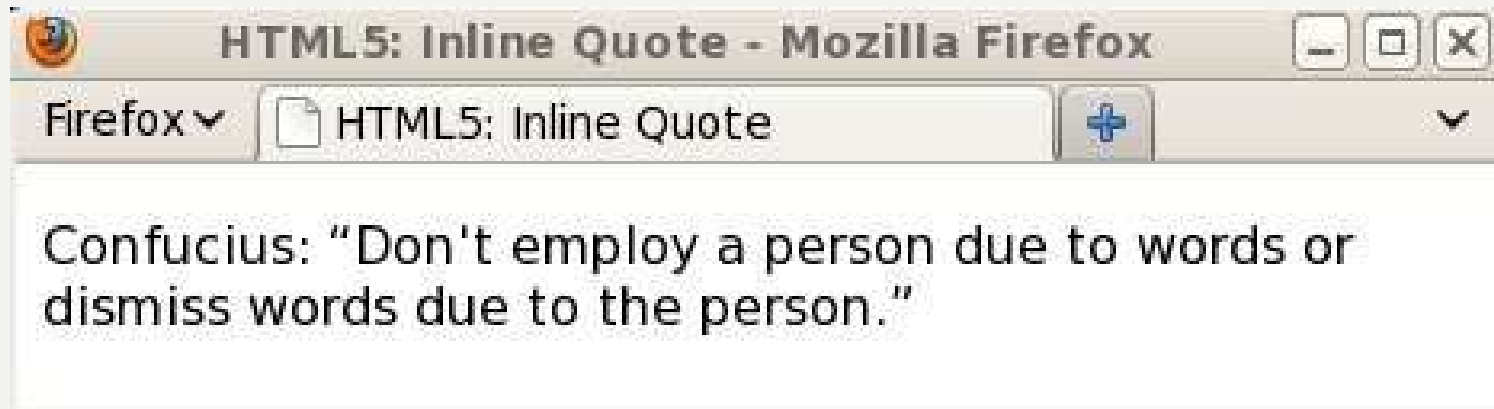
</blockquote>

**Demo: Ex:** Quote

<p>Confucius: <q>Don't employ a person due to words or  
dismiss words due to the person.</q></p>

**Demo: Ex:** InlineQuote

# INLINE QUOTE



```
<hr style="height: 4px; width: 50%;  
margin-left: auto; margin-right: auto" />
```

Demo: **Ex:** Hrule

# WHITE SPACE AND LINE WRAPPING

- HTML uses *white space* separates text into words.
- Words can be separated by one or more white-space characters but will only result in at most one rendered interword space.
- Tags do not break words. For example,

```
<p>The HTML<strong>5</strong> standard.</p>
```

# PREFORMATTED TEXT

- Text in a pre element is displayed in a fixed-width font, and it preserves both spaces and line breaks.

```
<figure style="width: 12em; background-color: cyan">
<pre>
    North

    West          East

    South
</pre></figure>
```



# Preformatted Text



Demo: **Ex:** Pre

# PHRASING ELEMENTS

- `a`: a link
- `br`: an explicit line break
- `cite`: a citation
- `em`: emphasis, usually displayed in italics
- `strong`: strong emphasis, usually displayed in boldface
- `mark`: stronger emphasis with highlighting
- `code`: computer code, usually displayed in a monospaced font
- `del`: deleted words displayed with a line through them
- `sub`: subscript (e.g., `x<sub>0</sub>`)
- `sup`: superscript (e.g., `x<sup>2</sup>`)
- `samp`: sample computer output

- `span`: a general phrasing element that can contain other phrasing (inline) elements, providing a simple way to attach presentation styles to enclosed elements; for example,  

```
<span style="font-weight: bold; color: blue">  
Important point</span>
```
- `var`: a variable
- `kbd`: keyboard text

# FORMATTED TIME

```
<time datetime="date_time">text</time>
```

```
<p>Fireworks start at <time datetime="2011-07-04T19:00">  
7pm on Independence Day</time></p>
```

```
<p>The final NASA space shuttle Atlantis  
launched on
```

```
<time datetime="2011-07-08T11:29-04:00">the  
morning of Friday, 08 July 2011</time>  
in Cape Canaveral, Florida USA.</p>
```

# PUBLICATION DATE

- Put a `time` element as child of `body` or child of the desired article.
- Give `datetime` a date string with optional time string.
- Add the attribute `pubdate="pubdate"`
- Note: The `time` element does not render as anything special in any of the major browsers.

```
<body>  
<time datetime="2012-07-07" pubdate="pubdate"></time>  
...  
</body>
```

# WEBPAGE PRESENTATION STYLES

Define the `style` attribute for an individual element.

```
<h1 style="color: darkgreen">The Green Earth Project</h1>
```

# STYLE **ATTRIBUTE**

General form:

```
style="property1:value1; property2:value2; . . . "
```

Foreground and Background Colors:

```
color: some_color
```

```
background-color: some_color
```

# TEXT ALIGNMENT

- `text-align: left`—lines are left justified
- `text-align: right`—lines are right justified
- `text-align: center`—lines are centered
- `text-align: justify`—lines are justified left and right



# FONT SIZES

```
<footer style="font-size: x-small">  
<p> ... </p>  
...  
<p> ... </p>  
</footer>
```

Demo: **Ex:** FontSize

# INDENTING CONTENT

To indent first line

```
<p style="text-indent: 3em"> ... </p>
```

To indent entire flow (block) element

```
margin-left: length
```

```
margin-right: length
```

```
<div style="margin-left: 5em; margin-right: 5em">
```

```
<p> ... </p>
```

```
</div>
```

# STYLE LENGTH UNITS

- `em`—the *font-size* of the current font
- `ex`—the *x-height* of the current font
- `ch`—the size of 0 (zero) of the current font

“Ems” (em): The “em” is a scalable unit that is used in web document media. An em is equal to the current font-size, for instance, if the font-size of the document is 12pt, 1em is equal to 12pt.

	<code>body { font-size: 100%; }</code>	<code>body { font-size: 120%; }</code>
<code>font-size: 1em</code>	The quick brown fox	The quick brown
<code>font-size: 12pt</code>	The quick brown fox	The quick brown fox
<code>font-size: 16px</code>	The quick brown fox	The quick brown fox
<code>font-size: 100%</code>	The quick brown fox	The quick brown

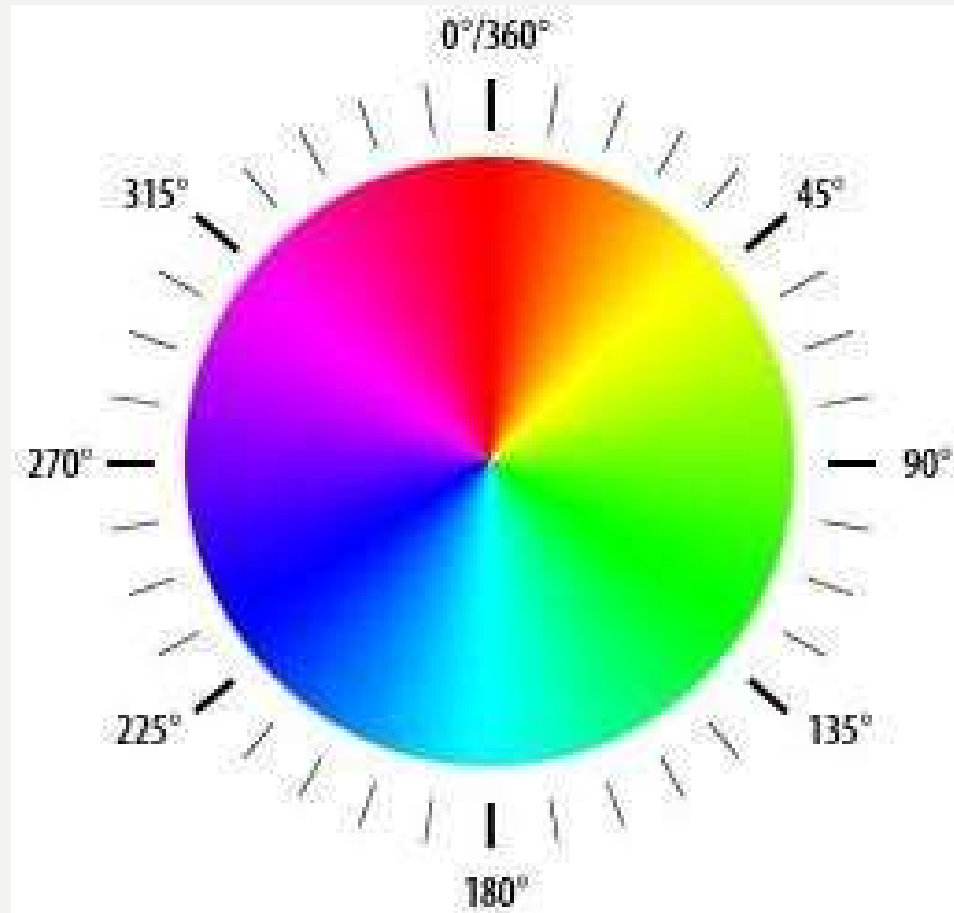
© KyleSchaeffer.com

# COLORS

Color values in style properties can be a color name such as `magenta` or `darkblue`. Currently there are about 150 color names defined in CSS. Color values can also be given in a number of standard notations, including RGB (red-green-blue) and HSL (hue-saturation-lightness):

1. `#rrggbb`—where the first two, middle two, and last two of the six hexadecimal digits specify red, green, and blue values, respectively (e.g., `#0ace9f`). This is 24-bit color.
2. `#rgb`—shorthand for the above notation when the first two, middle two, and last two digits are the same (e.g., `#03c` stands for `#0033cc`).
3. `rgb(r , g , b )`—where base-10 integers between 0 and 255 inclusive are used (e.g., `rgb(0,204,108)`). This is the decimal equivalent of notation 1.
4. `hsl(h , s%, l%)`—where *h* (in 0–360 degrees) indicates the hue on the color wheel.
5. [https://www.w3schools.com/colors/colors\\_rgb.asp](https://www.w3schools.com/colors/colors_rgb.asp)
6. [https://www.w3schools.com/colors/colors\\_hsl.asp](https://www.w3schools.com/colors/colors_hsl.asp)

# THE COLOR WHEEL



# FONT STYLES

font-family

font-style

font-variant

font-weight

font-size

font-family: Times

font-family: Arial, Helvetica, sans-serif

Demo: **Ex:** FontFamily

# SOME FONTS

Times	Arial	Helvetica
Courier	Monospace	

# GENERIC FONT FAMILIES

- serif—for example, Times
- sans-serif—for example, Arial or Helvetica
- cursive—for example, Zapf-Chancery
- fantasy—for example, Western
- monospace—for example, Courier

<http://wavian.com/font-list.html>

<https://websitesetup.org/web-safe-fonts-html-css/>



# FONT WEIGHT

`font-weight: normal`

`font-weight: bold`

`font-weight: bolder`

`font-weight: lighter`

# RELATIVE FONT SIZES

xx-small   x-small   small   medium   large  
x-large   xx-large

xx-small   x-small   small  
medium   large   x-large   xx-  
large

## Absolute Font Sizes

- pt (points; 1 pt = 1/72 in.)
- pc (picas; 1 pc = 12 pt)

# ITEMIZED LISTS

- Bullet list: The `ul` element provides an *unordered list* where the ordering of the items is unimportant. A `ul` is usually presented as a set of bulleted items.
- Ordered list: The `ol` element offers a *numbered list* where the ordering of the items is important. An `ol` is typically displayed as a sequence of enumerated items.
- Definition list: The `dl` element is handy for a *definition list* where each term (`<dt>`) is given a definition or description (`<dd>`).

```
<ul>
<li>Tropical Fruits
  <ol><li>Pineapple</li><li>Banana</li>
    <li>papaya</li></ol>
</li>
<li>Cereals
  <ol><li>Barley</li> <li>Rice</li> <li>Wheat</li></ol>
</li>
<li>Vegetables
  <ol><li>Broccoli</li> <li>Onion</li> <li>Yam</li></ol>
</li></ul>
```

Demo: **Ex:** List

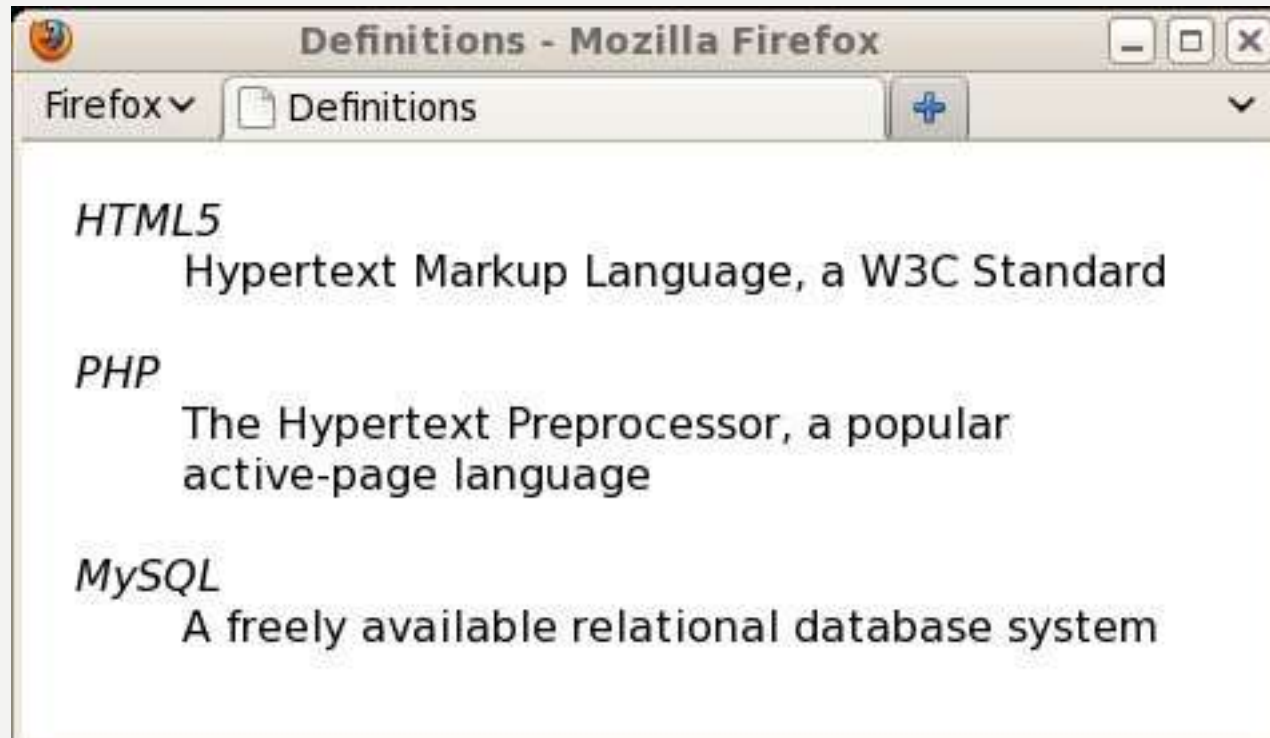
# LISTS



```
<dl>
  <dt style="font-style: italic">HTML5</dt>
  <dd>Hypertext Markup Language, a W3C Standard<br />
    <br /></dd>
  <dt style="font-style: italic">PHP</dt>
  <dd>The Hypertext Preprocessor, a popular
    active-page language <br /><br /></dd>
  <dt style="font-style: italic">MySQL</dt>
  <dd>A freely available relational database system</dd>
</dl>
```

Demo: **Ex:** Defs

# A DEFINITION LIST



# LIST STYLES

```
<ul style="list-style-type: circle"> ... </ul>
```

```
<ol style="list-style-type: upper-alpha"> ... </ol>
```

<p>The following list has inside positioning</p>

```
<ul style="list-style-position: inside">
```

```
  <li style="list-style-type: square; color: green">
```

```
    <span style="color: black">First item in the list  
    with a green square marker.</span></li>
```

```
  <li style="list-style-type: square; color: red">
```

```
    <span style="color: black">Second item in the list  
    with a red square marker.</span></li>
```

```
  <li style="list-style-type: square; color: blue">
```

```
    <span style="color: black">Third item in the list  
    with a blue square marker.</span></li>
```



`</ul>`

`list-style: circle inside`

Demo: **Ex:** MarkerStyle

# LINKS IN WEBPAGES

```
<a href="URL">anchor</a>
```

```
<a href="bio.html">Brief Bio</a>
```

```
<a href="http://www.w3.org/"> The W3C  
Consortium</a>
```

```
<a href="../../../pic/dragonfly.jpg" type="image/jpeg"  
  title="dragonfly.jpg">Picture of Dragonfly</a>
```

```
<a href="sound/cthd.mp3" type="audio/mpeg">  
  Tan Dun, Yo Yo Ma - Crouching Tiger,  
  Hidden Dragon - Theme</a>
```

# IN-PAGE LINKS

```
<h3 id="products">Our Quality Products</h3>
```

```
<a href="URL#products"> ... </a>
```

```
<article>
```

```
<nav><ul>
```

```
<li><a href="#product">Products</a></li>
```

```
<li><a href="#service">Services</a></li>
```

```
<li><a href="#testimonial">Testimonials</a></li>
```

```
</ul></nav>
```

```
<section>
```

```
<h3 id="product">Our Quality Products</h3>
```

```
...
```

```
</section><section>
```

```
<h3 id="service">Responsive Services</h3>
```

```
...
```

```
</article>
```

# SITE INTERNAL AND EXTERNAL LINKS

- Is clearly indicated as going off site. Often a webpage will include external links, such as references and advertisements, in an `aside` element displayed on the right-hand side of the page.
- Is displayed in a new browser window or tab so the visitor can come back by closing that new window or tab. A simple way is to use the attribute `target="_blank"` to cause the referenced page to display in a new window/tab:

```
<a href="http://www.w3.org/" target="_blank">  
The W3C Consortium</a>
```

# SITE ORGANIZATION AND NAVIGATION

- Organize the pages for a site into a hierarchy of files and directories (folders) stored on the hard disk of the server host. Avoid nonalphanumeric characters in file and directory names. Otherwise, the file name must be URL encoded before becoming part of a URL.
- Place the site entry page (usually, `index.html`) in the *server root* directory.
- Use subdirectories such as `images/`, `videos/`, `css/` (for style sheets), `js/` (for JavaScript code), `products/`, `services/`, `contractors/`, `members/`, and `affiliates/` to organize the site. The `index.html` page within each subdirectory is usually the lead page for that part of the site.
- Keep the organization simple and avoid using more than three

levels of subdirectory nesting.

- Design a navigation system that is clear, easy to use, and effective in getting visitors where they want to go in your site.
- Use relative URLs exclusively for linking within the site and make sure the link is in one of these forms:
  1. Relative to the *host page* itself (`href="file"` or `href="dir/file"`)
  2. Relative to the server root (`href="/path-to-file"`)