Introduction to HTML

1.1 What is HTML and Where Did it Come from?

The history might begin with the ARPANET of the late 1960s, jump quickly to the first public specification of the HTML by Tim Berners-Lee in 1991, and then to HTML's codification by the World Wide Web Consortium (better known as the W3C) in 1997. Some histories of HTML might also tell stories about the Netscape Navigator and Microsoft Internet Explorer of the early and mid-1990s, a time when intrepid developers working for the two browser manufacturers ignored the W3C and brought forward a variety of essential new tags (such as, for instance, the <table> tag), and features such as CSS and JavaScript, all of which have been essential to the growth and popularization of the web.

The publishing language used by the World Wide Web is HTML (from HyperText Markup Language).

HTML is defined as a markup language. A markup language is simply a way of annotating a document in such a way as to make the annotations distinct from the text being annotated. Markup languages such as HTML, Tex, XML, and XHTML allow users to control how text and visual elements will be laid out and displayed.

XHTML

Instead of growing HTML, the W3C turned its attention in the late 1990s to a new specification called XHTML 1.0, which was a version of HTML that used stricter XML (extensible markup language) syntax rules. The goal of XHTML with its strict rules was to make page rendering more predictable by forcing web authors to create web pages without syntax errors. To help web authors, two versions of XHTML were created: XHTML 1.0 Strict and XHTML 1.0 Transitional. The strict version was meant to be rendered by a browser using the strict syntax rules and tag support described by the W3C XHTML 1.0 Strict specification; the transitional recommendation is a more forgiving flavor of XHTML, and was meant to act as a temporary transition to the eventual global adoption of XHTML Strict.

HTML5

At around the same time the XHTML 2.0 specification was being developed, a group of developers at Opera and Mozilla formed the WHATWG (Web Hypertext Application Technology Working Group) group within the W3C. This group was not convinced that the W3C's embrace of XML and its abandonment of backwards-compatibility was the best way forward for the web. Thus the WHATWG charter introduced HTML5

There are three main aims to HTML5:

1. Specify unambiguously how browsers should deal with invalid markup.

2. Provide an open, nonproprietary programming framework (via JavaScript) for creating rich web applications.

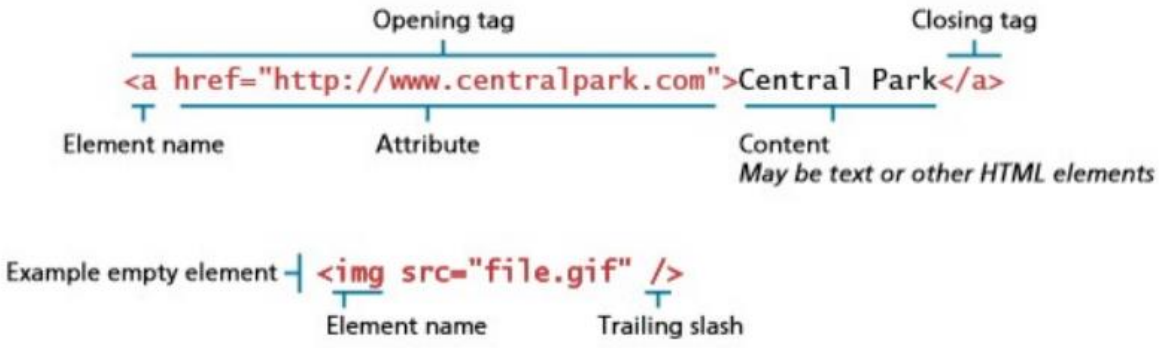
3. Be backwards compatible with the existing web.

1.2 HTML Syntax

Elements and Attributes

HTML documents are composed of textual content and HTML elements. An HTML element is identified in the HTML document by tags. A tag consists of the element name within angle brackets. The element name appears in both the beginning tag and the closing tag, which contains a forward slash followed by the element's name, again all enclosed within angle brackets. The closing tag acts like an off-switch for the on-switch that is the start tag.

HTML elements can also contain attributes. An HTML attribute is a name=va1ue pair that provides more information about the HTML element. In XHTML, attribute values had to be enclosed in quotes; in HTML5, the quotes are optional, though many web authors still maintain the practice of enclosing attribute values in quotes. Some HTML attributes expect a number for the value. These will just be the numeric value; they will never include the unit.



An empty element does not contain any text content; instead, it is an instruction to the browser to do something. Perhaps the most common empty element is +img>, the image element.

Nesting HTML elements

Often an HTML element will contain other HTML elements. In such a case, the container element is said to be a parent of the contained, or child, element. Any elements contained within the child are said to be descendants of the parent element; likewise, any given child element, may have a variety of ancestors.

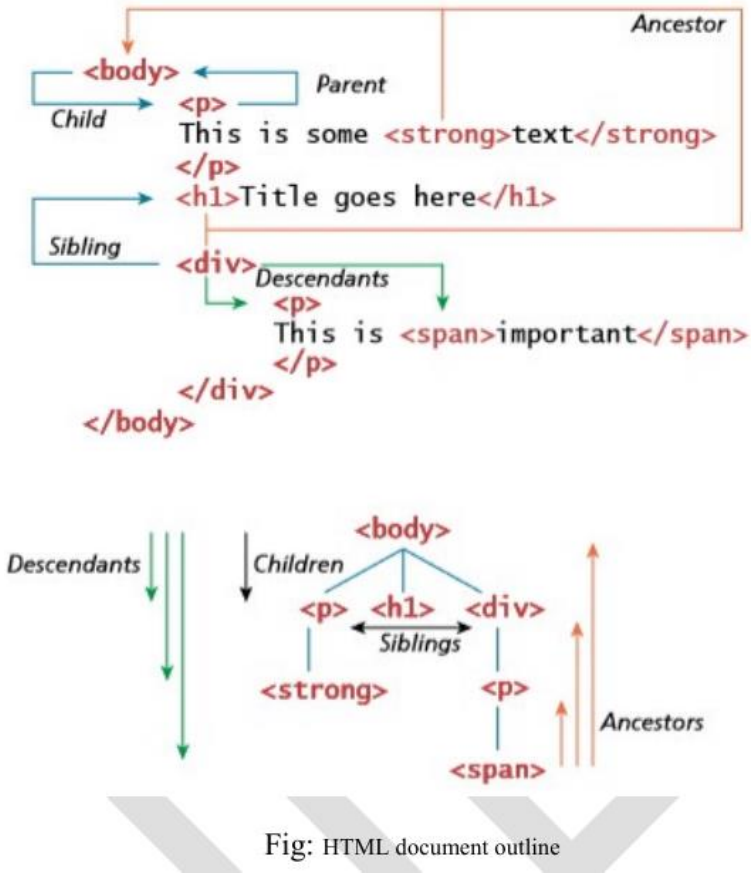


Fig: HTML document outline

1.3 Semantic Markup

HTML authors are often counseled to create semantic HTML documents. That is, an HTML document should not describe how to visually present content, but only describe its content's structural semantics or meaning.

Eliminating presentation-oriented markup and writing semantic HTML markup has a variety of important advantages:

• **Maintainability.** Semantic markup is easier to update and change than web pages that contain a great deal of presentation markup.

• **Faster.** Semantic web pages are typically quicker to author and faster to download.

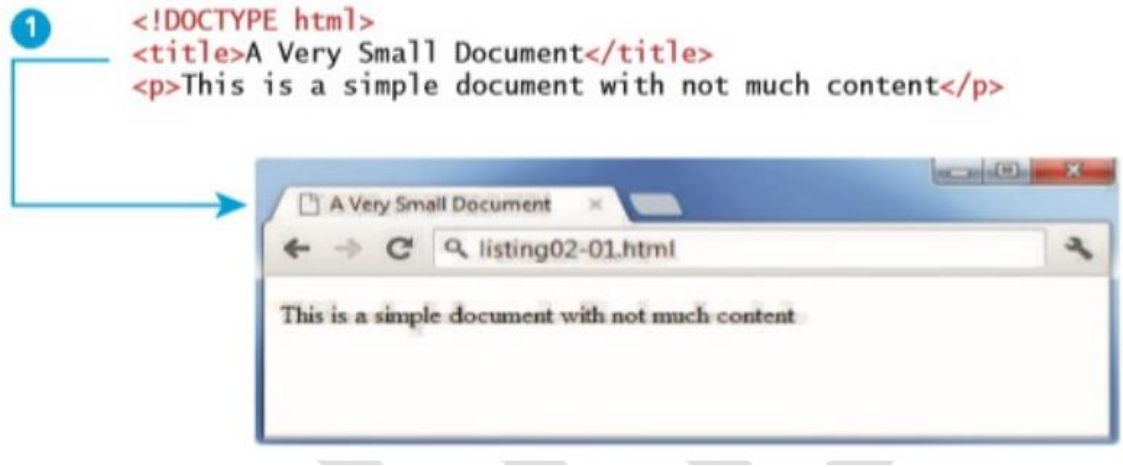
• **Accessibility.** Not all web users are able to view the content on web pages. Users with sight disabilities experience the web using voice reading software. Visiting a web page using voice reading software can be a very frustrating experience if the site does not use semantic markup

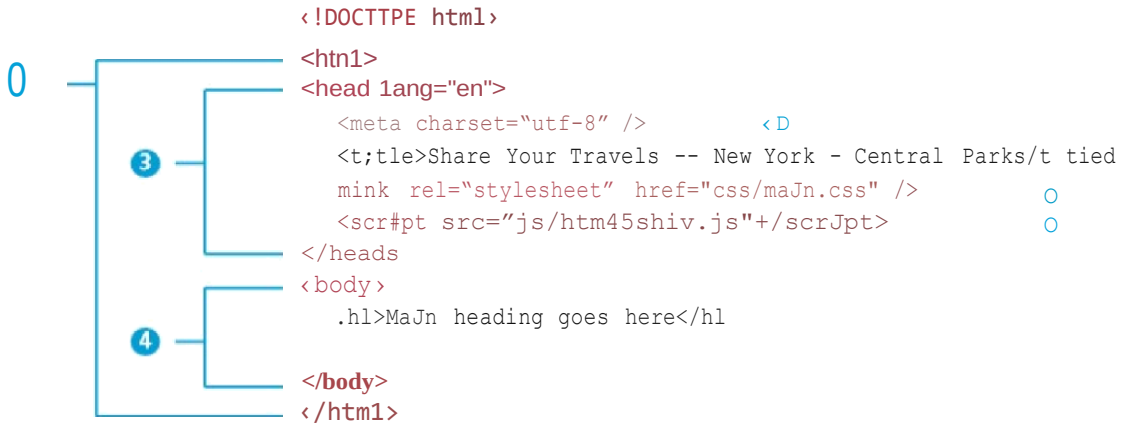
• **Search engine optimization**. For many site owners, the most important users of a website are the various search engine crawlers. These crawlers are automated programs that cross the web scanning sites for their content, which is then used for users' search queries. Semantic markup

provides better instructions for these crawlers: it tells them what things are important content on the site.

1.4 Structure of HTML documents

The <title> element is used to provide a broad description of the content. The title is not displayed within the browser window. Instead, the title is typically displayed by the browser in its window and/or tab. The title has some additional uses that are also important to know. The title is used by the browser for its bookmarks and its browser history list.





1. DOCTYPE

DOCTYPE (short for Document Type Definition) element, which tells the browser (or any other client software that is reading this HTML document) what type of document it is about to process.

2. Head and Body

The <htm1> element is sometimes called the **root** element as it contains all the other HTML elements in the document. Notice that it also has a **lang** attribute. This optional attribute tells the browser the natural language that is being used for textual content in the HTML document, which is English in this example.

HTML pages are divided into two sections: the head and the body, which correspond to the <head> and <body> elements. The head contains descriptive elements about the document, such as its title, any style sheets or JavaScript files it uses, and other types of meta information used by search engines and other programs. The body contains content (both HTML elements and regular text) that will be displayed by the browser.

<meta> element declares that the character encoding for the document is UTF-8. Character encoding refers to which character set standard is being used to encode the characters in the document.

1.5 Quick tour of HTML elements

- Headings

HTML provides six levels of heading (h1 through h6), with the higher heading number indicating a heading of less importance. Headings are also used by the browser to create a document outline for the page. Every web page has a document outline. It is also potentially used by web authors when they write JavaScript to manipulate elements in the document or when they use CSS to style different HTML elements.

- Paragraphs and Divisions

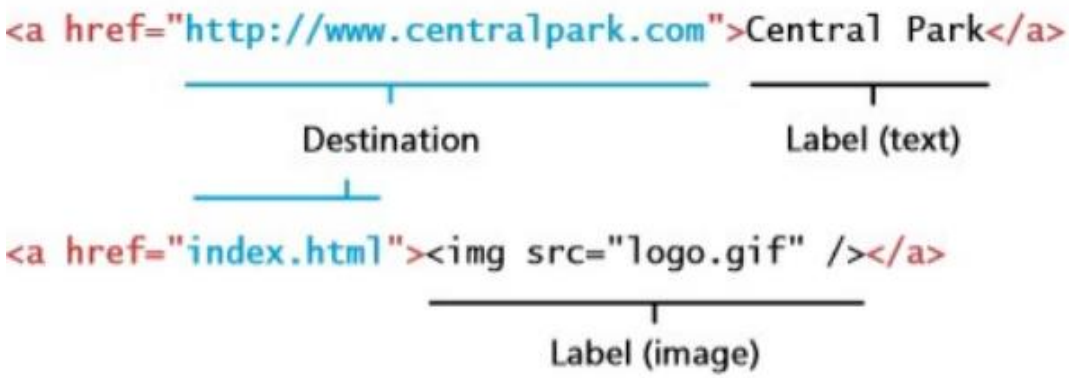
Paragraphs are represented by the <p> tag which is a container and can contain HTML and other inline HTML elements (the and <a> elements). The line break element
 forces a line break. It is suitable for text whose content belongs in a single paragraph but which must have specific line breaks: for example, addresses and poems.

The element is also a container element and is used to create a logical grouping of content (text and other HTML elements, including containers such as <p> and other elements). The element has no intrinsic presentation; it is frequently used in contemporary CSS-based layouts to mark out sections.

- Links

Links are an essential feature of all web pages. Links are created using the <a> element (the "a" stands for anchor). A link has two main parts: the destination and the label.

You can use the anchor element to create a wide range of links.



These include:

- Links to external sites (or to individual resources such as images or movies on an external site).
- Links to other pages or resources within the current site.
- Links to other places within the current page.
- Links to particular locations on another page (whether on the same site or on an external site).
- Links that are instructions to the browser to start the user's email program.
- Links that are instructions to the browser to execute a JavaScript function.
- Links that are instructions to the mobile browser to make a phone call.

- URL Relative Referencing

Whether we are constructing links with the had element, referencing images with the <img> element, or including external JavaScript or CSS files, we need to be able to successfully reference files within our site. This requires for so-called relative referencing.

When referencing a page or resource on an external site, a full absolute reference is required: that is, a complete URL with a protocol (typically, http://), the domain name, any paths, and then finally the file name of the desired resource. However, when referencing a resource that is on the same server as your HTML document, you can use briefer relative referencing. If the URL does not include the "http://" then the browser will request the current server for the file. If all the resources for the site reside within the same directory (also referred to as a folder), then you can reference those other resources simply via their file name.

However, most real-world sites contain too many files to put them all within a single directory. For these situations, a relative pathname is required along with the file name. The pathname tells the browser where to locate the file on the server. Pathnames on the web

follow Unix conventions. Forward slashes ("/") are used to separate directory names from each other and from file names. Double-periods ("..") are used to reference a directory "above" the current one in the directory tree.

The following table provides additional explanations and examples of the different types of URL referencing.

Relative Link Type	Example
1 Same Directory To link to a file within the same folder, simply use the file name.	To link to example.html from about.html (in Figure 2.17), use: <a href="example.html">
2 Child Directory To link to a file within a subdirectory, use the name of the subdirectory and a slash before the file name.	To link to logo.gif from about.html, use: <a href="images/logo.gif">
3 Grandchild/Descendant Directory To link to a file that is multiple subdirectories *below* the current one, construct the full path by including each subdirectory name (separated by slashes) before the file name.	To link to background.gif from about.html, use: <a href="css/images/background.gif">
4 Parent/Ancestor Directory Use "../" to reference a folder above the current one. If trying to reference a file several levels above the current one, simply string together multiple "../".	To link to about.html from index.html in members, use: <a href="../about.html"> To link to about.html from bio.html, use: <a href="../../about.html">
5 Sibling Directory Use "../" to move up to the appropriate level, and then use the same technique as for child or grandchild directories.	To link to about.html from index.html in members, use: <a href="../images/about.html"> To link to background.gif from bio.html, use: <a href="../../css/images/background.gif">
6 Root Reference An alternative approach for ancestor and sibling references is to use the so-called root reference approach. In this approach, begin the reference with the root reference (the "/") and then use the same technique as for child or grandchild directories. **Note that these will only work on the server! That is, they will not work when you test it out on your local machine.**	To link to about.html from bio.html, use: <a href="/about.html"> To link to background.gif from bio.html, use: <a href="/images/background.gif">
7 Default Document Web servers allow references to directory names without file names. In such a case, the web server will serve the default document, which is usually a file called index.html (Apache) or default.html (IIS). **Again, this will only generally work on the web server.**	To link to index.html in members from about.html, use either: <a href="members"> Or <a href="/members">

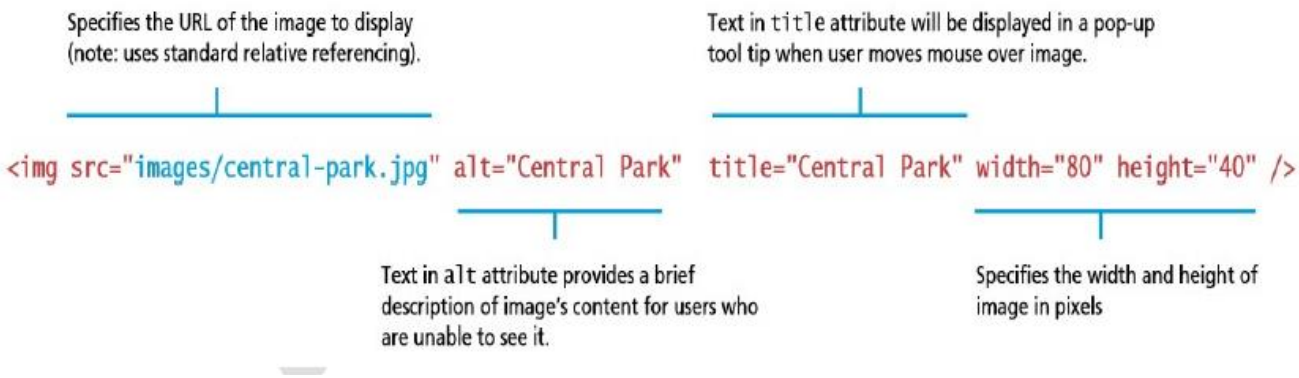
- **Inline text elements**

Inline elements do not disrupt the flow of text (i.e., cause a line break). HTML defines over 30 of these elements. Some of the most commonly used of these elements are:

Element	Description
<a>	Anchor used for hyperlinks.
<abbr>	An abbreviation
 	Line break
<cite>	Citation (i.e., a reference to another work).
<code>	Used for displaying code, such as markup or programming code.
	Emphasis
<mark>	For displaying highlighted text
<small>	For displaying the fine-print, i.e., "non-vital" text, such as copyright or legal notices.
	The inline equivalent of the element. It is generally used to mark text that will receive special formatting using CSS.
	For content that is strongly important.
<time>	For displaying time and date data

- **Images**

The <img> tag is the oldest method for displaying an image; it is not the only way. In fact, it is very common for images to be added to HTML elements via the background-image property in CSS



- **Character Entities**

These are special characters for symbols for which there is either no easy way to type them via a

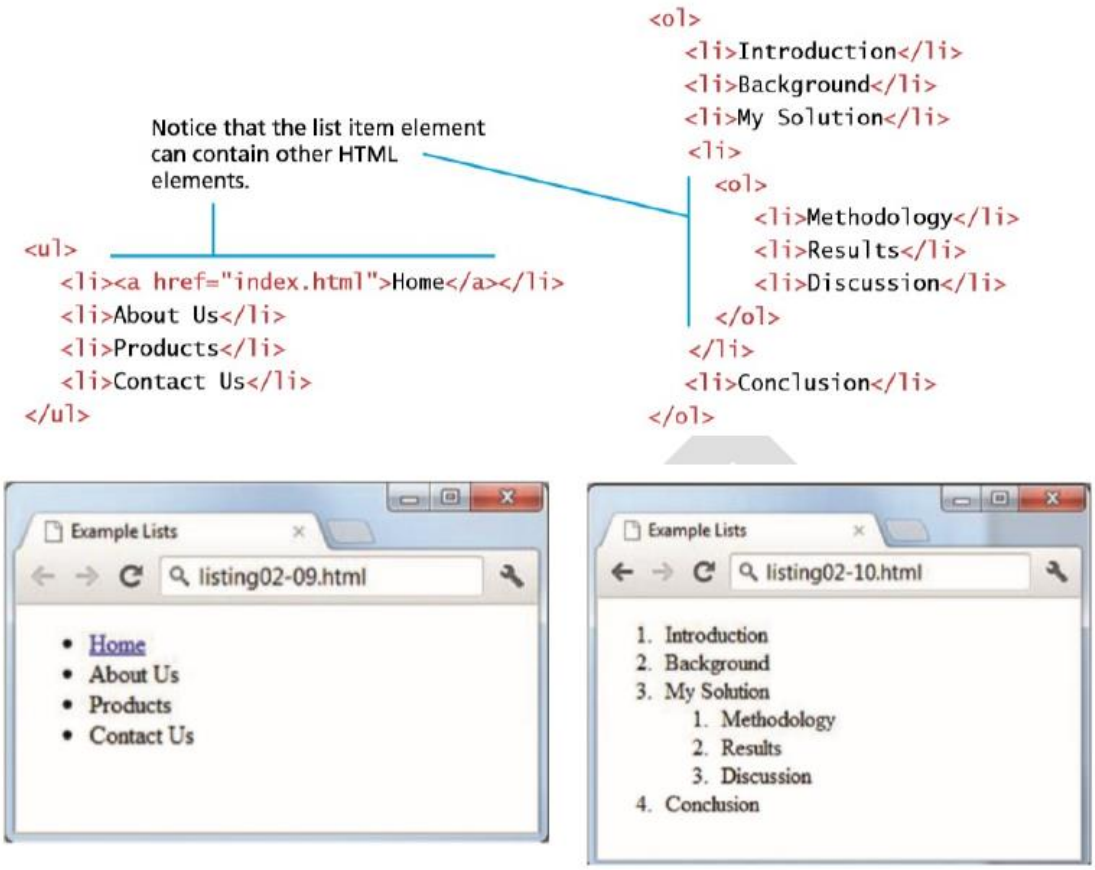
keyboard (such as the copyright symbol or accented characters) or which have a reserved meaning in HTML (for instance the "<" or ">" symbols). There are many HTML character entities. They can be used in an HTML document by using the entity name or the entity number.

Entity Name	Entity Number	Description
		Nonbreakable space. **The browser ignores multiple spaces in the source HTML file. If you need to display multiple spaces, you can do so using the nonbreakable space entity.**
<	<	Less than symbol ("<").
>	>	Greater than symbol (">").
©	©	The © copyright symbol.
€	€	The € euro symbol.
™	™	The ™ trademark symbol.
ü	ü	The ü—i.e., small u with umlaut mark.

- **Lists**

HTML provides three types of lists:

- **Unordered lists.** Collections of items in no particular order; these are by default rendered by the browser as a bulleted list. However, it is common in CSS to style unordered lists without the bullets. Unordered lists have become the conventional way to markup navigational menus.
- **Ordered lists**. Collections of items that have a set order; these are by default rendered by the browser as a numbered list.
- **Definition lists.** Collection of name and definition pairs. These tend to be used infrequently. Perhaps the most common example would be a FAQ list.



1.6 HTML5 Semantic Structure Elements

As HTML5 was being developed, researchers at Google and Opera had their search spiders examine millions of pages to see what were the most common id and class names. Their findings helped standardize the names of the new semantic block structuring elements in HTML5.

- **Header and Footer**

Most website pages have a recognizable header and footer section. Typically the header contains the site logo and title (and perhaps additional subtitles or taglines), horizontal navigation links, and perhaps one or two horizontal banners. The typical footer contains less important material, such as smaller text versions of the navigation, copyright notices, information about the site's privacy policy, and perhaps twitter feeds or links to other social sites. Both the HTML5 <header> and <footer> element can be used not only for page headers and footers but also for header and footer elements inside HTML5 container

- **Heading Groups**

It is not that unusual for a header to contain multiple headings in close proximity. The <hgroup> element can be used in such a circumstance to group them together within one container.

```
<header>
   <hgroup>
   <h1>Chapter Two: HTML 1</h1>
   <h2>An Introduction</h2>
   </hgroup>
</header>
<article>
   <hgroup>
   <h2>HTML5 Semantic Structure Elements</h2>
   <h3>Overview</h3>
   </hgroup>
</article>
```

- **Navigation**

The <nav> element represents a section of a page that contains links to other pages or to other parts within the same page. Like the other new HTML5 semantic elements, the browser does not apply any special presentation to the <nav> element.

```
<header>
   <img src="logo.gif" alt="logo" />
   <h1>Fundamentals of Web Development</h1>
   <nav role="navigation">
     <ul>
        <li><a href="index.html">Home</a></li>
        <li><a href="about.html">About Us</a></li>
        <li><a href="browse.html">Browse</a></li>
     </ul>
   </nav>
</header>
```

- **Article and Section**

The article <artic1e> element represents a section of content that forms an independent part of a document or site; for example, a magazine or newspaper article, or a blog entry.

The section <section> element represents a section of a document, typically with a title or heading.

- **Figures and Figure Captions**

```
<p>This photo was taken on October 22, 2011 with a Canon EOS 30D camera.</p>
<figure>
  <img src="images/central-park.jpg" alt="Central Park" /><br/>
  <figcaption>Conservatory Pond in Central Park</figcaption>
</figure>
<p>
```

- **Aside**

The <aside> element is similar to the <figure> element in that it is used for marking up content that is separate from the main content on the page. But while the <figure> element was used to indicate important information whose location on the page is somewhat unimportant, the <aside> element "represents a section of a page that consists of content that is tangentially related to the content around the aside element"

1.7 CSS: Introduction Cascading Style Sheets

CSS (Cascading Style Sheets) are the principal mechanism for web authors to modify the visual presentation of their web pages.

What is CSS?

CSS is a W3C standard for describing the appearance of HTML elements. Another common way to describe CSS's function is to say that CSS is used to define the presentation of HTML documents. With CSS, we can assign font properties, colors, sizes, borders, background images, and even position elements on the page. CSS can be added directly to any HTML element (via the style attribute), within the <head> element, or, most commonly, in a separate text file that contains only CSS.

Benefits of CSS

The benefits of CSS include:

- Improved control over formatting. The degree of formatting control in CSS is significantly better than that provided in HTML.
- Improved site maintainability. Websites become significantly more maintainable because all formatting can be centralized into one CSS file, or a small handful of them. This allows you to make site-wide visual modifications by changing a single file
- Improved accessibility. CSS-driven sites are more accessible. By keeping presentation out of the HTML, screen readers and other accessibility tools work better, thereby providing a significantly enriched experience for those reliant on accessibility tools.
- Improved page download speed. A site built using a centralized set of CSS files for all presentation will also be quicker to download because each individual HTML file will contain less style information and markup, and thus be smaller.
- Improved output flexibility. CSS can be used to adopt a page for different output media. This approach to CSS page design is often referred to as responsive design.

CSS Versions

The W3C decided to adopt CSS, and by the end of 1996 the CSS Level 1 Recommendation was published. A year later, the CSS Level 2 Recommendation (also more succinctly labeled simply as CSS2) was published. To make CSS3 more manageable for both browser manufacturers and web designers, the W3C has subdivided it into a variety of different CSS3 modules. So far the following CSS3 modules have made it to official W3C Recommendations: CSS Selectors, CSS Namespaces, CSS Media Queries, and CSS Color.

Browser Adoption

While Microsoft's Internet Explorer was an early champion of CSS (its IE3, released in 1996, was the first major browser to support CSS, and its IE5 for the Macintosh was the first browser to reach almost 100% CSS1 support in 2000), its later versions (especially IE5, IE6, and IE7) for Windows had uneven support for certain parts of CSS2. However, all browsers have not implemented parts of the CSS2 Recommendation. For this reason, CSS has a reputation for being a somewhat frustrating language.

1.8 CSS **SYNTAX**

A CSS document consists of one or more style rules. A rule consists of a selector that identifies the HTML element or elements that will be affected, followed by a series of **property: value** pairs (each pair is also called a declaration). The series of declarations is also called the declaration block. A declaration block can be together on a single line, or spread across multiple lines. The browser ignores white space (i.e., spaces, tabs, and returns) between your CSS rules so you can format the CSS however you want.

Each declaration is terminated with a semicolon. The semicolon for the last declaration in a block is in fact optional.

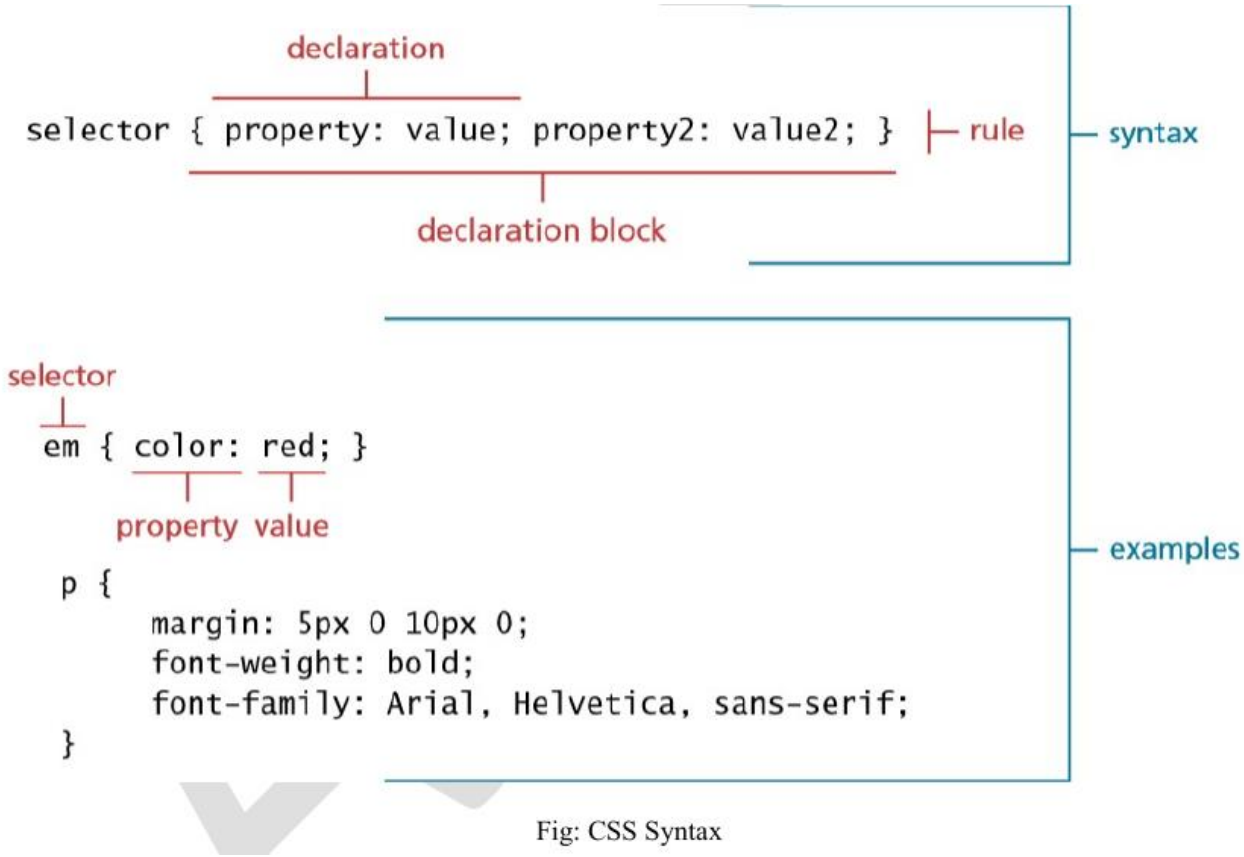


Fig: CSS Syntax

- **Selector:**

Every CSS rule begins with a selector. The selector identifies which element or elements in the HTML document will be affected by the declarations in the rule. Another way of thinking of selectors is that they are a pattern that is used by the browser to select the HTML elements that will receive the style.

- **Properties:**

Each individual CSS declaration must contain a property. These property names are predefined by the CSS standard. The CSS2.1 recommendation defines over a hundred different property names.

Property Type	Property
Fonts	font font-family font-size font-style font-weight @font-face
Text	letter-spacing line-height text-align text-decoration text-indent
Color and background	background background-color background-image background-position background-repeat color
Borders	border border-color border-width border-style border-top border-top-color border-top-width etc.

Property Type	Property
Spacing	padding padding-bottom, padding-left, padding-right, padding-top margin margin-bottom, margin-left, margin-right, margin-top
Sizing	height max-height max-width min-height min-width width
Layout	bottom, left, right, top clear display float overflow position visibility z-index
Lists	list-style list-style-image list-style-type

- **Values:**

Each CSS declaration also contains a value for a property. The unit of any given value is dependent upon the property. Some property values are from a predefined list of keywords.

1.9 LOCATION OF STYLES

CSS style rules can be located in three different locations. These three are not mutually exclusive, in that you could place your style rules in all three. In practice, however, web authors tend to place all of their style definitions in one (or more) external style sheet files.

CSS style rules can be located in three different locations.

1. Inline
2. Embedded
3. External

1. Inline Styles

Inline styles are style rules placed within an HTML element via the style attribute. An inline style only affects the element it is defined within and overrides any other style definitions for properties used in the inline style

Using inline styles is generally discouraged since they increase bandwidth and decrease maintainability (because presentation and content are intermixed and because it can be difficult to make consistent inline style changes across multiple files). Inline styles can, however, be handy for quickly testing out a style change.

```
<h1>Share Your Travels</h1>
<h2>style="font-size: 24pt"Description</h2>
...
<h2>style="font-size: 24pt; font-weight: bold;">Reviews</h2>
```

2. Embedded Style Sheet

Embedded style sheets (also called internal styles) are style rules placed within the <style> element (inside the <head> element of an HTML document).

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
   <style>
      h1 { font-size: 24pt; }
      h2 {
       font-size: 18pt;
       font-weight: bold;
      }
   </style>
</head>
<body>
   <h1>Share Your Travels</h1>
   <h2>New York - Central Park</h2>
   ...
```

3. External Style Sheet

External style sheets are style rules placed within a external text file with the .css extension. This is by far the most common place to locate style rules because it provides the best maintainability. When you make a change to an external style sheet, all HTML documents that reference that style sheet will automatically use the updated version. The browser is able to cache the external style sheet, which can improve the performance of the site as well. To reference an external style sheet, you must use a blink> element (within the <head> element)

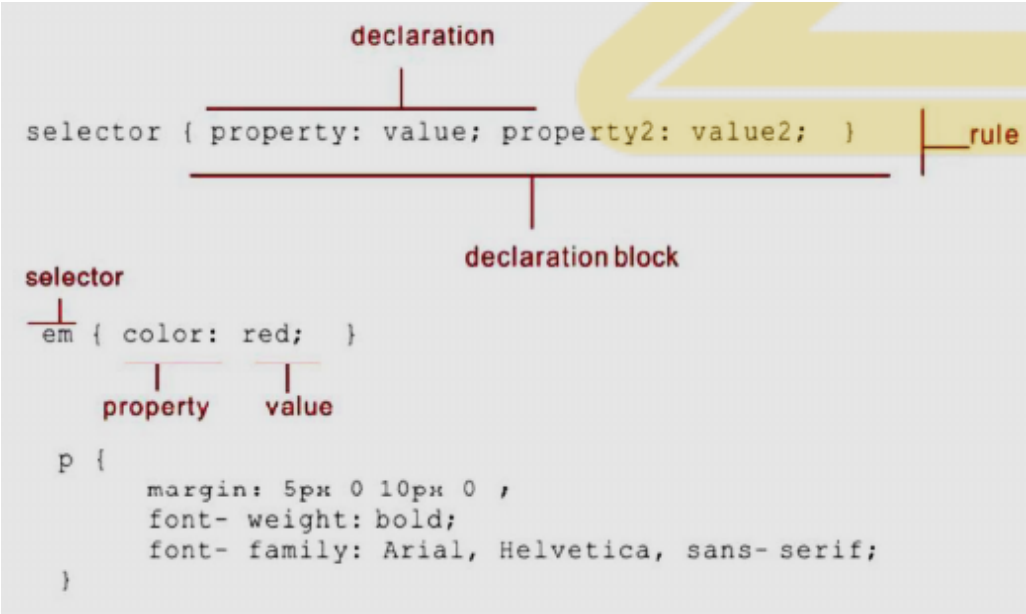
```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
   <link rel="stylesheet" href="styles.css" />
</head>
```

1.10 SELECTORS

When defining CSS rules, you will need to first use a selector to tell the browser which elements will be affected by the property values. CSS selectors allow you to select individual or multiple HTML elements.

- **Element Selectors**

Element selectors select all instances of a given HTML element. You can select all elements by using the **universal element selector**, which is the *** (asterisk)** character. You can select a group of elements by separating the different element names with commas. This is a sensible way to reduce the size and complexity of your CSS files, by combining multiple identical rules into a single rule.



- **Grouped Selectors**

```
/* commas allow you to group selectors */
p, div, aside {
   margin: 0;
   padding: 0;
}
/* the above single grouped selector is equivalent to the
   following: */
p {
   margin: 0;
   padding: 0;
}
div {
   margin: 0;
   padding: 0;
}
aside {
   margin: 0;
   padding: 0;
}
```

- You can select a group of elements by separating the different element names with commas.
- This is a sensible way to reduce the size and complexity of your CSS files, by combining multiple identical rules into a single rule.

- **Class Selectors**

A class selector allows you to simultaneously target different HTML elements regardless of their position in the document tree. If a series of HTML elements have been labeled with the same class attribute value, then you can target them for styling by using a class selector, which takes the form: period (.) followed by the class name.

```
<head>
   <title>Share Your Travels </title>
  <style>
         .first {
          font-style: italic;
          color: red;
         }
    </style>
</head>
<body>
   <h1 class="first">Reviews</h1>
   <div>
      <p class="first">By Ricardo on <time>September 15, 2015</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>
   <hr/>

   <div>
      <p class="first">By Susan on <time>October 1, 2015</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>
</body>
```

- **Id selector**

An id selector allows you to target a specific element by its id attribute regardless of its type or position. If an HTML element has been labeled with an id attribute then you can target it for styling by using an id selector, which takes the form: pound/hash (#) followed by the id name.

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels -- New York - Central Park</title>
  <style>
        #latestComment {
         font-style: italic;
         color: red;
        }
</style>
</head>
<body>
   <h1>Reviews</h1>
   <div id="latestComment">
      <p>By Ricardo on <time>September 15, 2015</time></p>
      <p>Easy on the HDR buddy.</p>
   </div>
   <hr/>
   <div>
      <p>By Susan on <time>October 1, 2015</time></p>
      <p>I love Central Park.</p>
   </div>
   <hr/>
</body>
```

- **Attribute Selector**

An attribute selector provides a way to select HTML elements either by the presence of an element attribute or by the value of an attribute. We can do this by using the following attribute selector:

[title] (... }

This will match any element in the document that has a title attribute.

```
<head lang="en">
   <meta charset="utf-8">
   <title>Share Your Travels</title>
    <style>
        [title] {
        cursor: help;
        padding-bottom: 3px;
        border-bottom: 2px dotted blue;
        text-decoration: none;
      }
  </style>
</head>
<body>
   <div>
      <img src="images/flags/CA.png" title="Canada Flag" />
      <h2><a href="countries.php?id=CA" title="see posts from Canada">
          Canada</a>
      </h2>
      <p>Canada is a North American country consisting of ... </p>
      <div>
          <img src="images/square/6114907897.jpg"
               title="At top of Sulphur Mountain" />
          <img src="images/square/6592317633.jpg"
               title="Grace Presbyterian Church" />
          <img src="images/square/6592914823.jpg"
               title="Calgary Downtown" />
    </div>
   </div>
</body>
```

Selector	Matches	Example
[]	A specific attribute.	`[title]` Matches any element with a title attribute
[=]	A specific attribute with a specific value.	`a[title="posts from this country"]` Matches any <a> element whose title attribute is exactly "posts from this country"
[~=]	A specific attribute whose value matches at least one of the words in a space-delimited list of words.	`[title~="Countries"]` Matches any `title` attribute that contains the word "Countries"
[^=]	A specific attribute whose value begins with a specified value.	`a[href^="mailto"]` Matches any <a> element whose `href` attribute begins with "mailto"
[*=]	A specific attribute whose value contains a substring.	`img[src*="flag"]` Matches any <img> element whose `src` attribute contains somewhere within it the text "flag"
[$=]	A specific attribute whose value ends with a specified value.	`a[href$=".pdf"]` Matches any <a> element whose `href` attribute ends with the text ".pdf"

- **Pseudo Element & Pseudo Class Selector**

A pseudo-element selector is a way to select something that does not exist explicitly as an element in the HTML document tree but which is still a recognizable selectable object. For instance, you can select the first line or first letter of any HTML element using a pseudo-element selector. A pseudo-class selector does apply to an HTML element, but targets either a particular state or, in CSS3, a variety of family relationships.

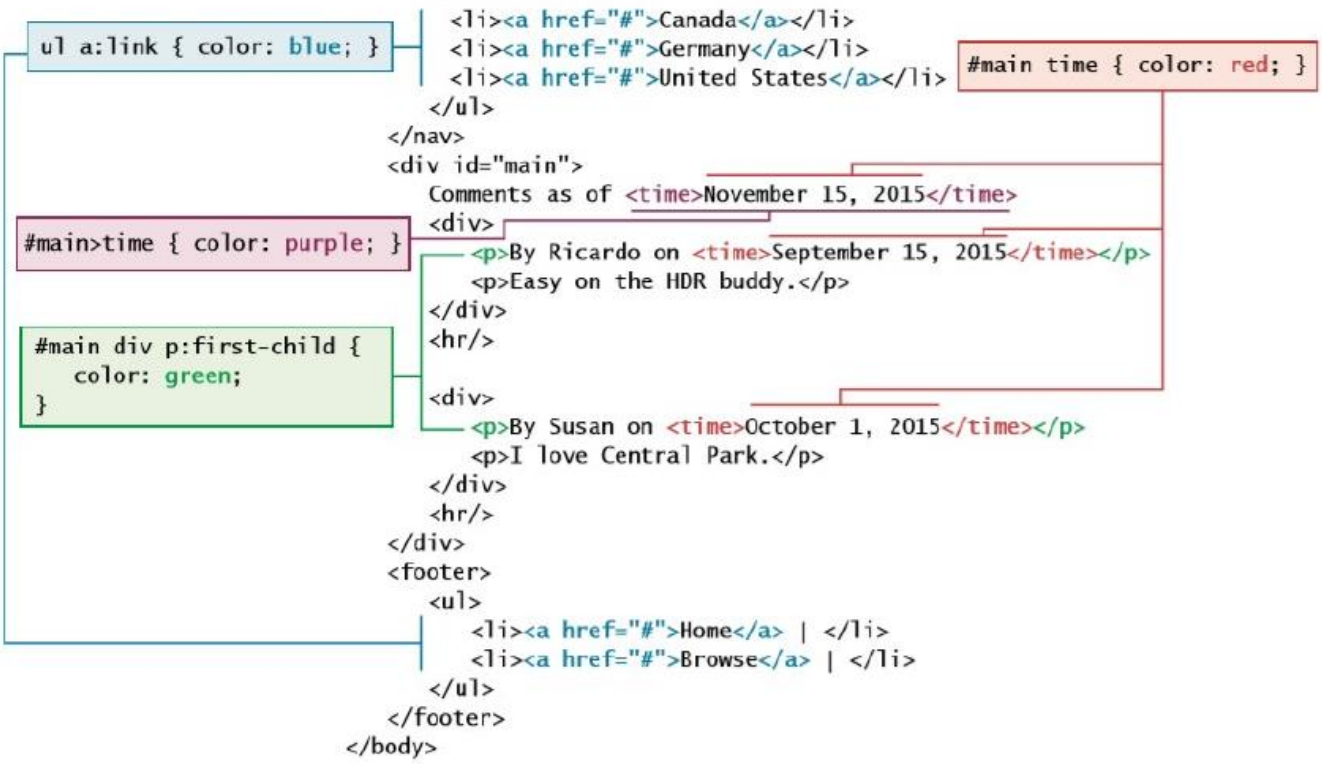
Selector	Type	Description
a:link	pseudo-class	Selects links that have not been visited
a:visited	pseudo-class	Selects links that have been visited
:focus	pseudo-class	Selects elements (such as text boxes or list boxes) that have the input focus.
:hover	pseudo-class	Selects elements that the mouse pointer is currently above.
:active	pseudo-class	Selects an element that is being activated by the user. A typical example is a link that is being clicked.
:checked	pseudo-class	Selects a form element that is currently checked. A typical example might be a radio button or a check box.
:first-child	pseudo-class	Selects an element that is the first child of its parent. A common use is to provide different styling to the first element in a list.
:first-letter	pseudo-element	Selects the first letter of an element. Useful for adding drop-caps to a paragraph.
:first-line	pseudo-element	Selects the first line of an element.

- **Contextual Selector**

A contextual selector (in CSS3 also called **combinators**) allows you to select elements based on their ancestors, descendants, or siblings. That is, it selects elements based on their context or their relation to other elements in the document tree.

A descendant selector matches all elements that are contained within another element. The character used to indicate descendant selection is the space character.

Selector	Matches	Example
Descendant	A specified element that is contained somewhere within another specified element.	`div p` Selects a <p> element that is contained somewhere within a element. That is, the <p> can be any descendant, not just a child.
Child	A specified element that is a direct child of the specified element.	`div>h2` Selects an <h2> element that is a child of a element.
Adjacent sibling	A specified element that is the next sibling (i.e., comes directly after) of the specified element.	`h3+p` Selects the first <p> after any <h3>.
General sibling	A specified element that shares the same parent as the specified element.	`h3~p` Selects all the <p> elements that share the same parent as the <h3>.



1.11 THE CASCADE. HOW STYLES INTERACT

There are three different types of style sheets: author-created, user-defined, and the default browser style sheet. As well, it is possible within an author-created stylesheet to define multiple rules for the same HTML element. For these reasons, CSS has a system to help the browser determine how to display elements when different style rules conflict. The "Cascade" in CSS refers to how conflicting rules are handled.

CSS uses the following cascade principles to help it deal with conflicts:

1. Inheritance,
2. Specificity, and
3. Location.

1. Inheritance

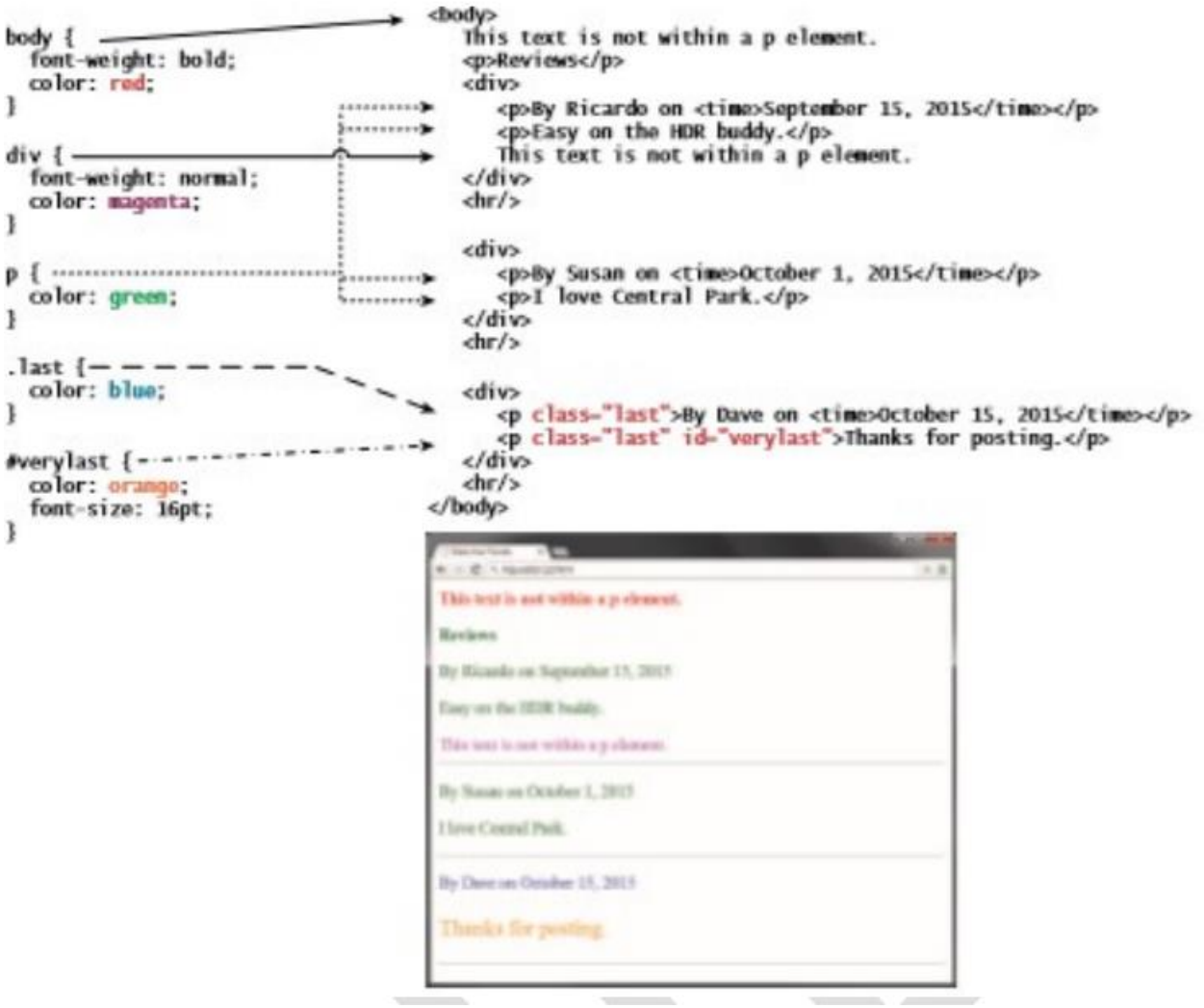
Inheritance is the first of these cascading principles. Many (but not all) CSS properties affect not only themselves but their descendants as well. Font, color, list, and text properties are inheritable; layout, sizing, border, background, and spacing properties are not.

2. Specificity

Specificity is how the browser determines which style rule takes precedence when more than one style rule could be applied to the same element. In CSS, the more specific the selector, the more it takes precedence (i.e., overrides the previous definition).
Another way to define specificity is by telling you how it works. The way that specificity works in the browser is that the browser assigns a weight to each style rule;

2.1 Specificity Algorithm

First count 1 if the declaration is from a 'style' attribute in the HTML, 0 otherwise (let that value =a).
Count the number of ID attributes in the selector (let that value =b).
Count the number of other attributes and pseudo-classes in the selector (let that value =c).
Count the number of element names and pseudo-elements in the selector (let that value= d).

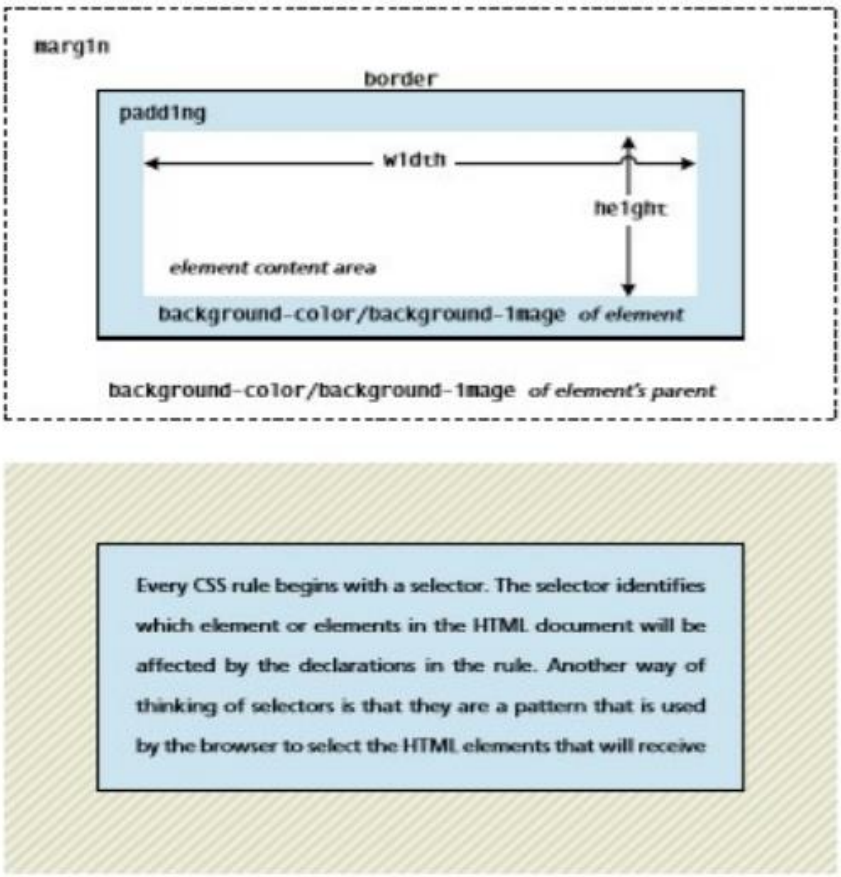


3. Location

Finally, when inheritance and specificity cannot determine style precedence, the principle of location will be used. The principle of location is that when rules have the same specificity, then the latest are given more weight. For instance, an inline style will override one defined in an external author style sheet or an embedded style sheet.

1.12 THE BOX MODEL

In CSS, all HTML elements exist within an element box as shown in the below diagram



- **Background**

As can be seen in the above figure, the background color or image of an element fills an element out to its border (if it has one, that is). In contemporary web design, it has become extremely common to use CSS to display purely presentational images (such as background gradients and patterns, decorative images, etc.) rather than using the <img> element. The below table lists the most common background properties

Property	**Description**
Background	A combined shorthand property that allows you to set multiple background values in one property.
background-attachment	Specifies whether the background image scrolls with the document (default) or remains fixed. Possible values are: fixed, scroll.
background-image	Specifies the background image (which is generally a jpeg, gif, or png file) for the element
background-position	Specifies where on the element the background image will be placed. Some possible values include: bottom, center, left, and right.

background-repeat	Determines whether the background image will be repeated, i.e to create a tiled look
background-color	Sets the background color of the element
background-size	this property lets you modify the size of the background image

- **Borders**

Borders provide a way to visually separate elements. You can put borders around all four sides of an element, or just one, two, or three of the sides.

Property	Description
border	A combined shorthand property that allows you to set the style, width, and color of a border in one property. The order is important and must be: `border-style border-width border-color`
border-style	Specifies the line type of the border. Possible values are: `solid, dotted, dashed, double, groove, ridge, inset, and outset.`
border-width	The width of the border in a unit (but not percents). A variety of keywords (`thin`, `medium`, etc.) are also supported.
border-color	The color of the border in a color unit.
border-radius	The radius of a rounded corner.
border-image	The URL of an image to use as a border.

- **Margins and Padding**

Margins and padding are essential properties for adding white space to a web page, which can help differentiate one element from another.

- **Box Dimensions**

Since the width and the height only refer to the size of the content area, the total size of an element is equal to the size of its content area plus the sum of its padding, borders, and margins.

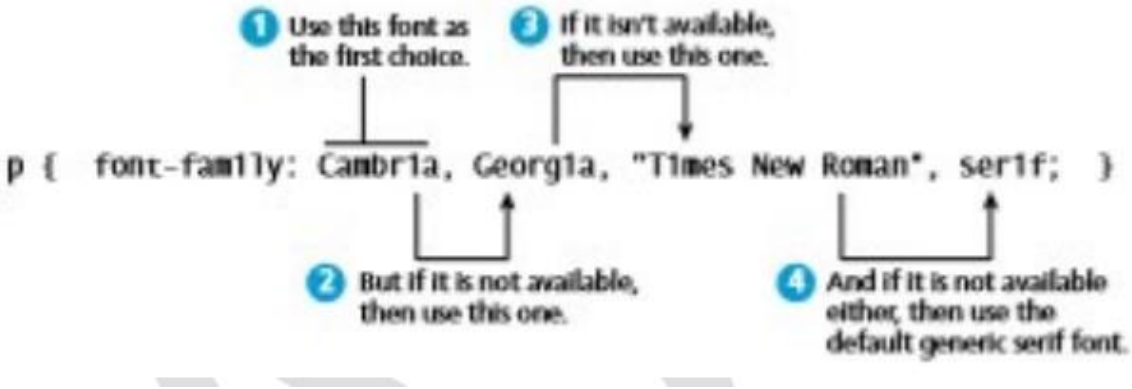
1.13 CSS TEXT STYLING

CSS provides two types of properties that affect text. The first we call font properties because they affect the font and its appearance. The second type of CSS text properties are referred to here as paragraph properties since they affect the text in a similar way no matter which font is being used.

- **Font Family**

The first of these problems involves specifying the font family. A word processor on a desktop machine can make use of any font that is installed on the computer; browsers are no different. However, just because a given font is available on the web developer's computer, it does not mean that that same font will be available for all users who view the site. For this reason, it is conventional to supply a so-called web font stack, that is, a series of alternate fonts to use in case the original font choice is not on the user's computer.

Property	Description
`font`	A combined shorthand property that allows you to set the family, style, size, variant, and weight in one property. While you do not have to specify each property, you must include at a minimum the font size and font family. In addtion, the order is important and must be: `style weight variant size font-family`
`font-family`	Specifies the typeface/font (or generic font family) to use. More than one can be specified.
`font-size`	The size of the font in one of the measurement units.
`font-style`	Specifies whether `italic`, `oblique` (i.e., skewed by the browser rather than a true italic), or `normal`.
`font-variant`	Specifies either `small-caps` text or none (i.e., regular text).
`font-weight`	Specifies either `normal`, `bold`, `bolder`, `lighter`, or a value between 100 and 900 in multiples of 100, where larger number represents weightier (i.e., bolder) text.



- **Font Size**

So while sizing with pixels provides precise control, if we wish to create web layouts that work well on different devices, we should learn to use relative units such as em units or percentages for our font sizes. CSS3 now supports a new relative measure, the rem (for root em unit). This unit is always relative to the size of the root element (i.e., the <html> element).

- **Paragraph Properties**

Just as there are properties that affect the font in CSS, there are also a range of CSS properties that affect text independently of the font.

Property	Description
letter-spacing	Adjusts the space between letters. Can be the value normal or a length unit.
line-height	Specifies the space between baselines (equivalent to leading in a desktop publishing program). The default value is normal, but can be set to any length unit. Can also be set via the shorthand font property.
list-style-image	Specifies the URL of an image to use as the marker for unordered lists.
list-style-type	Selects the marker type to use for ordered and unordered lists. Often set to none to remove markers when the list is a navigational menu or a input form.
text-align	Aligns the text horizontally in a container element in a similar way as a word processor. Possible values are left, right, center, and justify.
text-decoration	Specifies whether the text will have lines below, through, or over it. Possible values are: none, underline, overline, line-through, and blink. Hyperlinks by default have this property set to underline.
text-direction	Specifies the direction of the text, left-to-right (ltr) or right-to-left (rtl).
text-indent	Indents the first line of a paragraph by a specific amount.
text-shadow	A new CSS3 property that can be used to add a drop shadow to a text. Not yet supported in IE9.
text-transform	Changes the capitalization of text. Possible values are none, capitalize, lowercase, and uppercase.
vertical-align	Aligns the text vertically in a container element. Most common values are: top, bottom, and middle.
word-spacing	Adjusts the space between words. Can be the value normal or a length unit.