

# XHTML/CSS Review

## XHTML

An xhtml element tag in its simplest form:

```
<body></body>
```

Tags/Elements always have an opening tag, `<body>` and a end tag, `</body>`. End tags ALWAYS have the `'/'`.

*Exceptions to this are:*

- » `<meta / >` - Used for invisible information about a page
- » `<img />` - Used for inserting images
- » `<br />` - Used for creating line breaks

*Notice there is only ONE tag that has the `'/'` at the end and before the `'>'`.*

**Commonly used tags, you should be familiar with:**

- » `<html></html>` - tags for an html page - THESE ARE REQUIRED!
- » `<head></head>` - the tags allowing for heading information on a page - THESE ARE REQUIRED!
- » `<title></title>` - the tags that identify the title that will be displayed for the page
- » `<body></body>` - the tags that frame the body content for a page - THESE ARE REQUIRED!
- » `<ul></ul>` - tag for unordered lists
- » `<li></li>` - tag for list items
- » `<h1></h1>`- tag used for headings (1,2,3,4,5,6)
- » `<p></p>`- tag for paragraphs
- » `<strong></strong>` - tags used for bold type
- » `<em></em>` - tag used for italic type

**Attributes and Values.**

*For example:*

```
<body bgcolor="#FFFFFF"></body>
```

The body tag has been expanded upon to include the attribute `"bgcolor="` and the value of that attribute `"#FFFFFF"` (NOTE: All attribute values in html/xhtml must be surround by `"quotes"`, and all colors must be preceded by `"#"` ).

# CSS

---

## Style Types and The Cascade

In CSS, there are different types of styles: Browser, Author, and User. When it comes to the Cascade, in cascading style sheets, Author styles over-ride a browser style, and User styles over-ride all.

### *Browser Styles*

The style that is default and inherit in the browser itself. If an element is not styled, it will revert to the inherit browser style by default (i.e blue links with underlines).

### *Author Styles*

Author styles are styles created by the web designer/developer to layout the design/look-and-feel of the web site. There are several different ways to use author styles. *Author styles over-ride browser styles. NOTE: The closer a style is to the actual content, the greater specificity it has and thus can over-ride a previously declared style.*

#### Inline

The CSS is written inline, into the html code itself, for each element being styled. *These are closet to the content and will over-ride all other author styles and even user styles.*

```
<h1 style="color:#000000">Example</h1>
```

#### Embedded

The CSS is placed within the `<head>` portion of the html document, and is surrounded by a `<style>` tag.

```
<style type="text/css">
  h1{ color:#000000;}
</style>
```

#### Linked (External)

The CSS is located in a separate css file, and is linked to an html file using the `<link>` tag

```
<link href="sample.css" rel="Stylesheet" type="text/css" />
```

There is also the option when linking, to link to an 'Alternate Stylesheet

```
<link href="other.css" rel="Alternate Stylesheet" type="text/css" />
```

#### Imported (External)

The CSS is located in a separate css file. But this time, it is being imported using the `@import` and `url("")` tag, which is placed between a style tag.

```
<style type="text/css">
  @import url("sample.css");
</style>
```

The other way to import a css file, is directly into a css file. This is done using the same `@import` and `url("")` tag, but must be placed first at the top of the css document This allows you to modularize a sites css.

```
@import url("sample.css");
h1{ color:#000000;}
```

## User Styles

These are styles that are applied by the user with their own custom style sheet. These are generally changes to typography to improve legibility for the user, but sometimes users remove all graphic/design elements so the user is left only with content. User styles can over-ride any/all author styles.

### The anatomy of a CSS rule:

```
h1 { color: #000000; }
```

In the example above:

- » `h1` is known as the selector - refers to the XHTML tags or attributes to relate the rule to, instructs the browser to apply the rule to all `h1` elements in the XHTML document
- » The selector is followed by a curly bracket `{`
- » `color` is the property - characteristic of the selector, instructs the browser to change the color of `h1` elements to the value specified
- » A property is followed by a colon `:`
- » `#000000` is the value - definition of the property, changes the color of all `h1` elements to `#000000`
- » Values are followed by a semi-colon `;`
- » At the end of a rule, there is always a closing curly bracket `}`

Notice that there are no quotes around the "value" like in the html code and there are no '<>' around the `h1` tag

### How to write a CSS rule selector:

Know the difference between XHTML and CSS selectors:

- » `body` - applies to the XHTML (mark up) element `<body>`
- » `#name` - applies to an ID attribute tag in html called name (html code: `id="name"`)
- » `.name` - applies to an CLASS attribute tag in html called name (html code: `class="name"`)
- » `h1, h2, h3, h4, h5, h6` - listing contextual (tag) selector which applies to all `h#` tags without needing to define separate rules for each
- » `#header p` - Combined ID and contextual (tag) selector applies to all `p` tags inside of the item "header"
- » `table#firsttable` - Combined ID and contextual (tag) selector applies to the table with the ID "firsttable"
- » `.typearea p` - Combined class and contextual (tag) selector applies to all `p` tags inside of the item(s) "typearea"
- » `p.bodytype` - Combined class and contextual (tag) selector applies to all the `p` tag(s) with the class "bodytype"
- » `#header p strong` - Combining rules beyond just two items; applies to all strong tagged type inside `p` tags inside of the item "header"

### Pseudo-Class Selectors

Unlike a regular selector which exists physically in your html, a pseudo-class selector relates to your html, but does not exist in the html. It is used in conjunction with a regular selector and is noted using a colon ':' and are based off of the 'state' of an html element (i.e. hover, visited). Example:

*a:link*  
*a:visited*  
*a:hover*  
*a:active*

When styling links, the order LVHA (Link, Visited, Hover, Active), must be used to work in all browsers

*a:link, a:visited, a:hover, a:active*

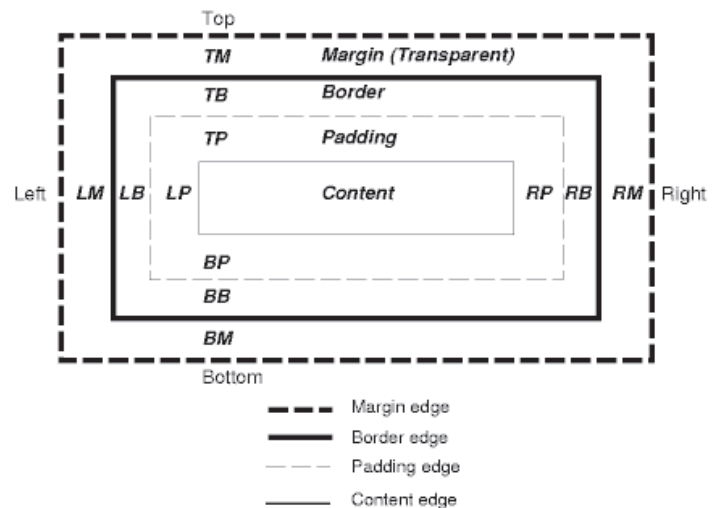
## Box versus Inline

### Main Box Types

- » Inline - *a, strong, span, img*, are by default styled with *display: inline* property
- » Block - *div, ul, ol*, are by default styled with *display: block* property
- » None - Useful for hiding things; CSS styled with *display: none* property

## The Box Model

Each box has a content area (e.g., text, an image, etc.) and optional surrounding padding, border, and margin areas; the size of each area is specified by properties defined below. The following diagram shows how these areas relate and the terminology used to refer to pieces of margin, border, and padding. The margin, border, and padding can be broken down into left, right, top, and bottom segments (e.g., in the diagram, "LM" for left margin, "RP" for right padding, "TB" for top border, etc.).



When a width is explicitly specified for any block-level element it should determine only the width of the content within the box, with the padding, borders and margins added afterwards. However, Internet Explorer incorrectly includes the padding and borders within the specified width, resulting in a wider box when displayed.

[http://en.wikipedia.org/wiki/Internet\\_Explorer\\_box\\_model\\_bug](http://en.wikipedia.org/wiki/Internet_Explorer_box_model_bug)