Client-side Technologies

Eng. Niveen Nasr El-Den
SD & Gaming CoE
iTi

Day 2

HTML cont.

The Mother Tongue of The Browser

<iframe>

- Inline or “floating” frames allow opening new pages inside main page.
- It provides a window that could be placed anywhere within an existing, non frame-based page.

```
<iframe src=“http://www.gamingegypt.com"
        height="200" width="200"
        frameborder="1">
      <p>Your browser does not support iframes</p>
</iframe>
```

Example!

Tables

Table

Food Categorization

vegetables		Fruits	
Name	Color	Name	Color
tomato	red	apple	yellow
Cucumber	dark green		green
carrot	orange		red

HTML Tables

- Tables represent tabular data
  - A table consists of one or several rows
  - Each row has one or more columns
- Table rows split into three semantic sections: header, body and footer
  - `<thead>` denotes table header and contains `<th>` elements, instead of `<td>` elements
  - `<tfoot>` denotes table footer but comes before the `<tbody>` tag
  - Last comes the body data `<tbody>` denotes collection of table rows that contain the very data

Table Tags

Tag	Description
<table>	Defines a table.
<caption>	Defines a table caption. Provides a means for labeling the table's content. Used once per table and must immediately follow the table start tag.
<th>	Defines a header cell in a table
<tr>	Defines a row in a table
<td>	Defines a cell in a table
<thead>	Groups the header content in a table. By default, a thead will not affect the display of the table in any way.
<tbody>	Groups the body content in a table
<tfoot>	Groups the footer content in a table

Using of <table>,<tr> & <td> Tags

- Graphical tables are enclosed within a two-sided **<table>** tag that identifies the start and ending of the table structure.
- Each row of the table is indicated using a two-sided **<tr>** (for table row).
- Within each table row, a two-sided **<td>** (for table data) tag indicates the presence of individual table cells.
- **<td>** can contain nested tables (tables within tables)

Columns Within a Table

- HTML does not provide a tag for table columns.
- In the original HTML specifications, the number of columns is determined by how many cells are inserted within each row.
  - i.e. if a table have four `<td>` tags in each row, then it has four columns.

The Table Syntax

- This creates a table with two rows and two columns.

```
<table>
    <tr>
        <td> First Cell </td>
        <td> Second Cell </td>
    </tr>
    <tr>
        <td> Third Cell </td>
        <td> Fourth Cell </td>
    </tr>
</table>
```

two rows

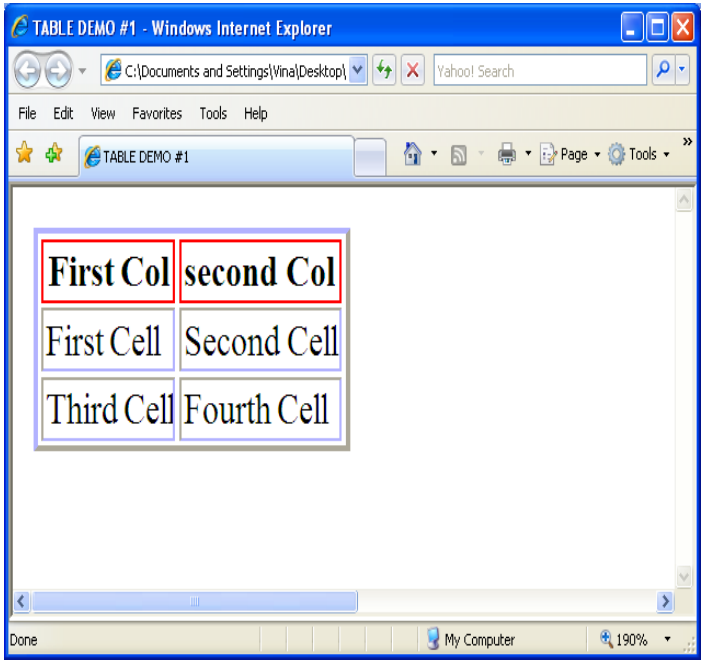
First Cell	Second Cell
Third Cell	Fourth Cell

two columns

Example!

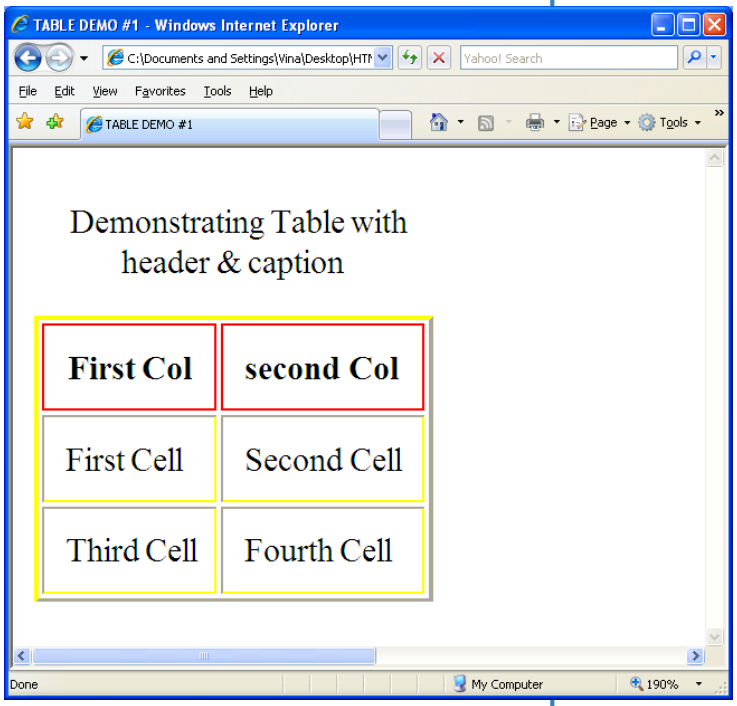
Adding Headings to Table

```
<table border="2"  bordercolor="#b2b2ff">
    <tr bordercolor="red">
        <th>First Col</th>
        <th>second Col</th>
    </tr>
    <tr>
        <td> First Cell </td>
        <td> Second Cell </td>
    </tr>
    <tr>
        <td> Third Cell </td>
        <td> Fourth Cell </td>
    </tr>
</table>
```



Adding <caption> to Table

```
<table  border="2"  bordercolor="yellow" cellpadding=10 >
      <caption>Demonstrating Table with header & caption</caption>
   <tr bordercolor="red">
        <th>First Col</th>
        <th>second Col</th>
   </tr>
   <tr>
        <td> First Cell </td>
        <td> Second Cell </td>
    </tr>
    <tr>
        <td> Third Cell </td>
        <td> Fourth Cell </td>
    </tr>
</table>
```

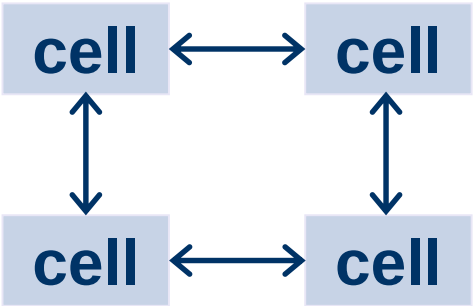


The <table> Tag Attributes

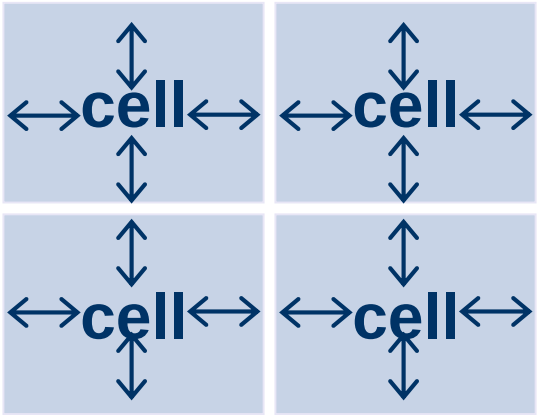
Attribute	Description
bgcolor	like with the body tag, this sets the background color of any item. It works with tables as well and will be seen as the default color.
width	determines the width of the table and isn't required. If you don't put this in, the table will automatically adjust in terms of width.
height	it's mainly for the height and how high you want the table to be. Not required at all and will adjust on its own if not with a set height.
border	determines the borders from outside the table to the table cells and isn't required. By default, browsers display tables without table borders, i.e. its default value is 0.
cellpadding	it's the distance of the cell wall from the contents.
cellspacing	sets the distance between cells and isn't required. Again, it will do that automatically if not set.
bordercolor	to set the color of the border.

Cell Spacing Vs. Cell Padding

- **Cell Spacing**
  Defines empty spaces between cells



- **Cell Padding**
  Defines empty spaces around cell content



Example!

The <tr> Tag Attributes

Attribute	Value	Description
align	Left Right Center justify	Aligns the content in a cell
bgcolor	*rgb(x,x,x)* *#xxxxxx* *colorname*	Specifies a background color for a cell.
valign	Top Middle Bottom baseline	Vertical aligns the content in a cell

The <th> & <td> Tags Attributes

Attribute	Value	Description
align	left |right | center | justify	Aligns the content in a cell
bgcolor	*rgb(x,x,x)* *#xxxxxx* *colorname*	Specifies a background color for a cell.
colspan	*Number*	Sets the number of columns a cell should span
height	*pixels* | *%*	Sets the height of a cell
rowspan	*Number*	Sets the number of rows a cell should span
valign	top | middle | bottom |baseline	Vertical aligns the content in a cell
width	*pixels* | *%*	Specifies the width of a cell

Defining Cell and Column Sizes

- To set the width of an individual cell, add the **width** attribute to either the **`<td>`** or **`<th>`** tags.

- The syntax is: **`width="value"`**
  - ***value*** can be expressed either in pixels or as a percentage of the table width
    - e.g. a width value of 30% displays a cell that is 30% of the total width of the table

- The **height** attribute can be used in the **`<td>`** or **`<th>`** tags to set the height of individual cells.

Spanning Rows & Columns

- To merge several cells into one, you need to create a **spanning cell**.
- A spanning cell is a cell that occupies more than one row or column in a table.
- Spanning cells are created by inserting the **rowspan** and **colspan** attribute in a **<td>** or **<th>** tag.
- The syntax for these attributes is: **rowspan="value" colspan="value"**
  - ***value*** is the number of rows or columns that the cell spans in the table

Column and Row Span

- colspan defines how many columns the cell occupies
- rowspan defines how many rows the cell occupies

Cell[1,1]	Cell[2,1]	
Cell[1,2]	Cell[2,2]	Cell[3,2]
Cell[1,3]		Cell[2,3]

```
<table>
    <tr>
        <td>Cell[1,1]</td>
        <td colspan="2">Cell[2,1]</td>
    </tr>
    <tr>
        <td>Cell[1,2]</td>
        <td rowspan="2">Cell[2,2]</td>
        <td>Cell[3,2]</td>
    </tr>
    <tr>
        <td>Cell[1,3]</td>
        <td>Cell[2,3]</td>
    </tr>
</table>
```

Example of Spanning Cells

Food Categorization

vegetables		Fruits	
Name	Color	Name	Color
tomato	red	apple	yellow
Cucumber	dark green		green
carrot	orange		red

this cell spans two columns

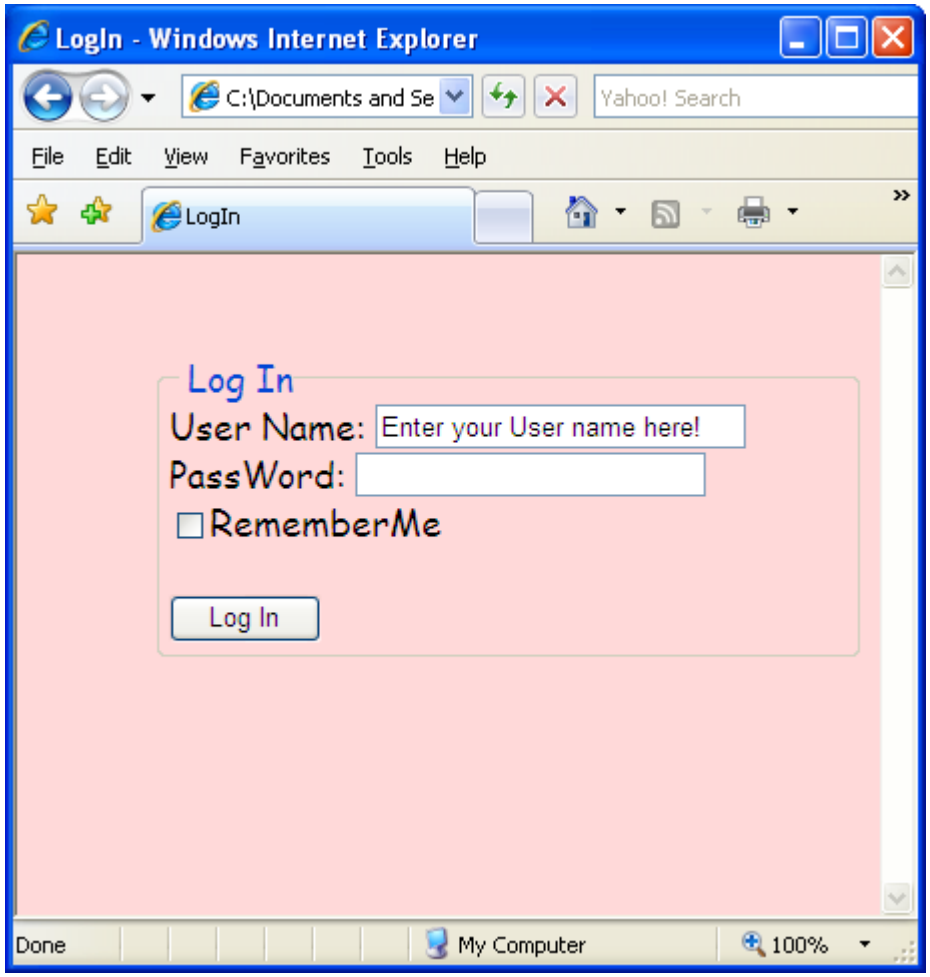
This cell spans three rows

Designing a Page Layout with Tables

- HTML tables are most often used to define the layout of an entire Web page.
- If you want to design a page that displays text in newspaper style columns, or separates the page into distinct sections, you'll find tables an essential and useful tool.

Forms

Sample Form Design



HTML Forms

```
<form>
    <!-- Here goes form fields and HTML -->
</form>
```

- Forms are the primary method for gathering data from site visitors
- <form> Main Attributes
  - **action=address**
    - Specifies the URL to which the form submission is sent to.
  - **method=post** or **method=get**
    - Specifies how to send form-data.

Form Fields

- A <form> can contain <input> elements presenting the following controls:
  - Text field
  - Password field
  - Hidden field
  - Check box
  - File
  - Submit button
  - Reset button
  - Ordinary button
  - Image button
  - Radio button
  - etc..
- Other controls:
  - Multi-line textarea
  - Drop-down menu

Form Fieldset

- <fieldset> is used to enclose a group of related form fields together.
- The <legend> is the fieldset's title.
- Example:

```
<form method="post" action="main.html">
  <fieldset>
    <legend>Client Details</legend>
    Name:<input type="text" id="Name" />
    Password:<input type="password" id="Pswd" />
  </fieldset>
</form>
```

Form Labels

- Form labels are used to associate an explanatory text to a form field using the field's ID.
- Clicking on a label focuses its associated field (checkboxes are toggled, radio buttons are checked)
- Example

```
<form>
    <label for="fn">First Name</label>
    <input type="text" id="fn" />
</form>
```

Navigation Fields

- tabindex attribute define a sequence that users follow when they use the Tab key to navigate through a page.

- *access keys* allow easier navigation by assigning a *keyboard shortcut* to a link. It can be used on **any** HTML element

Navigation Fields
(accesskey attribute)

Browser	Shortcut
Internet Explorer	[Alt] + *accesskey*
Chrome	
Safari	
Firefox	[Alt] [Shift] + *accesskey*
Opera 15 or newer Opera 12.1 or older	[Alt] + *accesskey* [Shift] [Esc] + *accesskey*

- if more than one element has the same access key differs:
  - IE, Firefox: The next element with the pressed access key will be activated
  - Chrome, Safari: The last element with the pressed access key will be activated
  - Opera: The first element with the pressed access key will be activated

<input> Field Attributes

- type
- size
- maxlength
- name
- id
- value
- tabindex: Specifies the tab order of an element.
- etc.

```
<input type="text" size="25" value="Enter your name!"/>
```

- Note:
  - Image buttons have the same effect as submit buttons with src, width, height attributes

<textarea> Field Attributes

- rows
- cols
- name
- tabindex
- etc..

```
<textarea cols="40" rows="5" name="myname">
    Now we are inside the area - which is nice.
</textarea>
```

Drop-Down Menu Tags

- <select> Attributes
  - name
  - size
  - multiple
- <option> Attributes
  - selected
  - value

```
<select>
    <option>Milk</option>
    <option>Coffee</option>
    <option>Tea</option>
</select>
```

- <optgroup> Attributes
  - label
  - disabled

```
<select>
    <optgroup label="Africa">
        <option>Egypt</option>
        <option>Sudan</option>
    </optgroup>
    <optgroup label="Europe">
        <option>France</option>
        <option>Russia</option>
    </optgroup>
</select>
```

XHTML

XHTML January 26, 2000.

- XHTML stands for “Extensible Hyper Text Mark-up Language”.
- XHTML is a new and more well-structured way of writing HTML.
- XHTML Versions
  - XHTML 1.0 became a W3C on January 26, 2000.
  - XHTML 1.1 became a W3C on May 31, 2001.
  - XHTML5 is undergoing development as of September 2009, as part of the HTML5 specification.
- XHTML consists of all the elements in HTML 4.01, combined with the strict syntax of XML.

The Most Important Differences

- XHTML elements must be **properly nested**.
- XHTML elements must always be **closed** even empty elements.
- XHTML elements must be in **lowercase**.
- XHTML documents must have **only one root element**.
- Some More XHTML Syntax Rules
  - Attribute names must be in **lower case**.
  - Attribute values must be **quoted**.
  - Attribute minimization is **forbidden**.
    - e.g. checked="checked".
  - The id attribute **replaces** the name attribute.
  - The XHTML DTD defines **mandatory** elements.
    - i.e. <!DOCTYPE html> Is Mandatory

<!doctype html>

- It is not an HTML tag
- It is an instruction to tell the web browser about what version of HTML the page is written in.
- It should always be the first item at the top of all your HTML files.
- It has no end tag.
- Browsers use a DOCTYPE in the beginning of the document to decide whether to handle it in
  - quirks mode or
  - standards mode.
- To ensure that your page uses full standards mode, make sure that your page has a DOCTYPE

documents must be marked up correctly & "well-formed"

HTML Online References

- www.w3schools.com
- www.echoecho.com
- www.tutorialspoint.com
- www.quackit.com
- www.htmlcodetutorial.com
- www.htmlquick.com
- www.htmldog.com
- https://developer.mozilla.org/en-US/docs/Web/HTML/Element
- www.tutorialehtml.com/en/index.php

Cascading Style Sheets cont.

the sister technology to HTML that is used to style your web pages

Cascading Order

- "Cascading " reflects the way styles are applied to the elements in a document, because style declarations cascade down to elements from many origins.
- Styles will be applied to HTML in the following order:
    1. Browser default
    2. External style sheet
    3. Internal style sheet (in head)
    4. Inline style
- When styles conflict, the "nearest" (most recently applied) style wins.

Example of Cascading Order

- **External Style sheet**

```
h3  {     color: red;
      text-align: left;
      font-size: 8pt    }
```

- **Internal Style sheet**

```
h3  {     text-align: right;
      font-size: 20pt;
      text-decoration: underline

      }
```

- **Resultant attributes**

```
color: red;
text-align: right;
font-size: 20pt;
text-decoration: underline
```

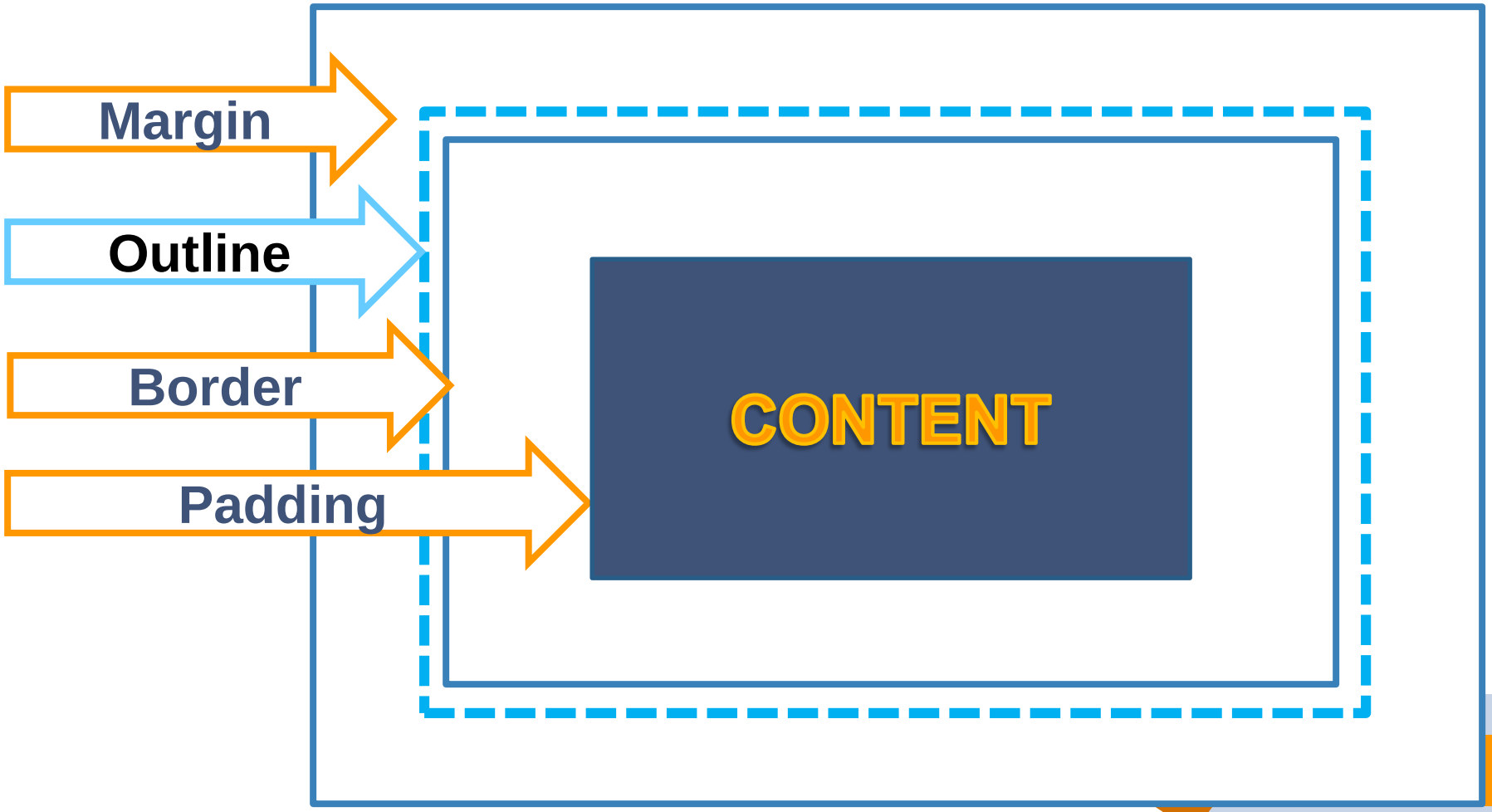
Example!

Box Model

- All HTML elements can be considered as boxes.
- The Box Model allows us to place a border around elements and space elements in relation to other elements.

- The Box Model consists of:
  - margins,
  - borders,
  - padding, and
  - the actual content.

Box Model

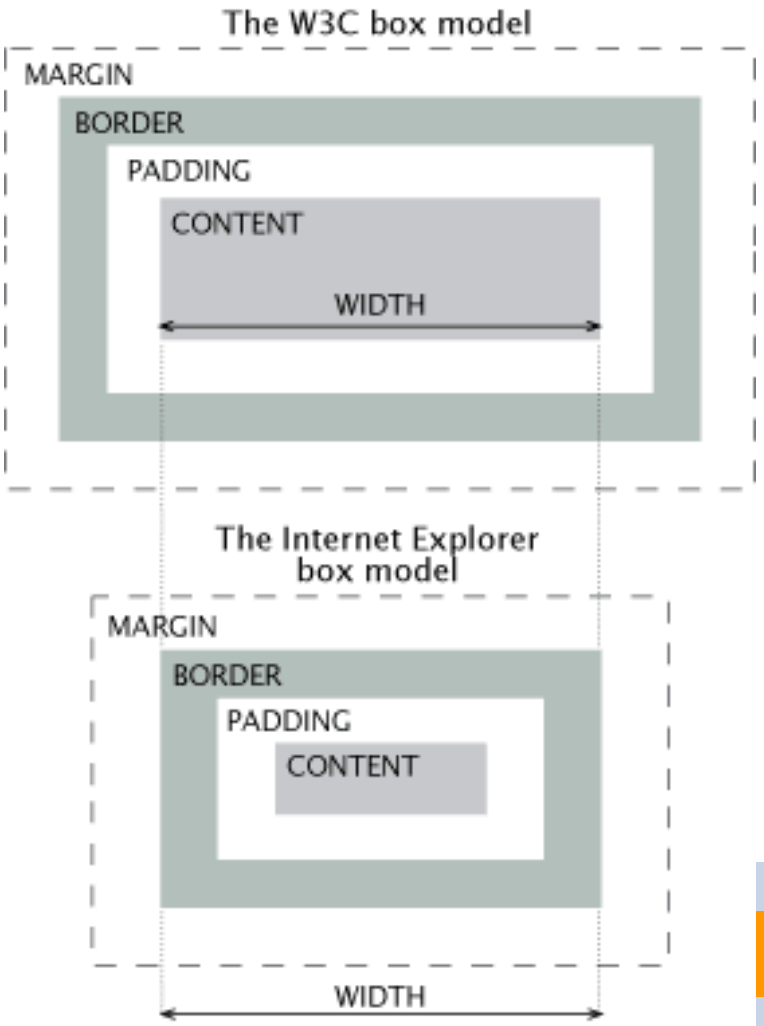


Quirks mode vs. Standards mode

- quirks mode
  - layout emulates nonstandard behavior in Navigator 4 and Internet Explorer 5 for Windows that is required not to break existing content on the Web.
- standards mode.
  - the behavior is (hopefully) the behavior described by the HTML and CSS specifications.

IE Quirks Mode

- When using quirks mode, Internet Explorer violates the box model standard



Borders, Padding, and Margins

CSS Property	Values
border-style:style border-top-style:style border-right-style:style border-bottom-style:style border-left-style:style	**Sets the *style* of a border surrounding a page element.** **It must be used if using any border property** **The *style* can be applied to all borders (border-style, borderStyle) or to selected borders. Style types can be** **dashed** **dotted** **double** **groove** **inset** **none** **outset** **ridge** **solid**

Borders, Padding, and Margins

CSS Property	Values
border-width:*width* **border-top-width**:*width* **border-right-width**:*width* **border-bottom-width**:*width* **border-left-width**:*width*	**Sets the *width* of a border surrounding a page element.** **The *width* can be applied to all borders (border-width, borderWidth) or to selected borders. Widths can be** **thin** **medium** **thick** ***n*px**

Borders, Padding, and Margins

CSS Property	Values
border-color:*color* border-top-color:***color*** border-right-color:***color*** border-bottom-color:***color*** border-left-color:***color***	**Sets the *color* of a border surrounding a page element.** **The *color* can be applied all borders (border-color, borderColor) or to selected borders. The *color* is specified as a color name, hexadecimal value, or RGB value.**

Borders, Padding, and Margins

CSS Property	Values
border-radius : px **border-radius : px px px px** border-top-left-radius: px; border-top-right-radius: px; border-bottom-right-radius: px; border-bottom-left-radius: px;	**Used for displaying round corners surrounding the element** **Sets the round corners for either the top left or top right, or bottom right, or bottom left corner of an element.** **Use the shorthand property border-radius to set the radius for the four corners**

Borders, Padding, and Margins

CSS Property	Values
border:***style width color***	**Border styles, widths, and colors can be set with the single border specification by coding these values separated by a blank space:** **border:solid 1px red** **border="solid 1px red"**

Borders, Padding, and Margins

CSS Property	Values
padding: px padding-top : px padding-right: px padding-bottom : px padding-left : px	**Properties that control box's padding. (the area between its content and its border.)** **Sets the padding for either the top or right, or bottom, or left side of an element.** **Use the shorthand property padding to set the padding for the four sides** px, pt, em

Borders, Padding, and Margins

CSS Property	Values
margin: px margin-top : px margin-right: px margin-bottom : px margin-left : px	**Properties that control box's margin. (the area outside its border.)** **Sets the margins for either the top or right, or bottom, or left side of an element.** **Use the shorthand property margin to set the margins for the four sides** px, pt, em

CSS Selectors

- Several types of selectors are defined for use when implementing Style Sheets:
    1. ~~Simple Basic Selectors~~
    2. Attribute selectors
    3. Combinators
    4. Pseudo-Classes
    5. Pseudo-Elements

- A selector can contain a chain of one or more simple selectors separated by combinators, optionally followed by attribute selectors, ID selectors, or pseudo-classes. but it can contain only one pseudo-element, which must be appended to the last simple selector in the chain

2. Attribute Selector

- Allows you to specify rules that match attributes defined in the source document.
- Syntax :

```
Input [type="button"]  {background-color: blue}
```

  - element [att] { property:value}
    - Match when the element sets the "att" attribute, whatever the value of the attribute.
  - element [att = “val”] {property: value}
    - Match when the element's "att" attribute value is exactly "val".
  - element [att^ = “val”] {property: value}
    - Match when the element's "att" attribute value starts with "val".
  - element [att$= “val”] {property: value}
    - Match when the element's "att" attribute value ends with "val".
  - element [att* = “val”] {property: value}
    - Match when the element's "att" attribute value contains "val".

3. Combinators

- A selector can contain more than one simple selector. Between the simple selectors, we can include a combinator.
- There are 4 different combinators.
  - descendant selector
    - matches an element that's a descendant of a specified element
  - child selector
    - selects an element that's the immediate child of a specified element
  - adjacent sibling selector
    - selects an element that's an adjacent sibling to a specified element
  - general sibling selector (CSS3)
    - selects an element that's a sibling to a specific element

3.1 Descendant/Contextual Selector

- Used when we want selectors to match an element that is the descendant of another element in the document tree.

```
<h1>This headline is <em>very</em>
important</h1>
```

- Example:

```
H1 { color: red }
EM { color: red }
```

This headline is *very* important

Although the intention of these rules is to add emphasis to text by changing its color, the effect will be lost .

To solve this:

```
H1 { color: red }
EM { color: red }
H1 EM { color: blue }
```

This headline is *very* important

3.1 Descendant/Contextual Selector

```
<style>
      p.myClass
     { color: green;}
</style>
<body>
     <p class="myClass">
     It's new Day..
     < em >
         Hello Everybody!!
     </ em >
     </p>
</body>
```

```
<style>
     p em
     { color: green; }
</style>
<body>
     <p>
     It's new Day..
     < em >
        Hello
        Everybody!!
     </em>
     </p>
</body>
```

Example!

3.2 Child Selector

- Matches when an element is the child of some element.
- A child selector is made up of two or more selectors separated by ">“.
- Example:

```
BODY > P {text-align: right }
```

  - The following rule sets the style of all P elements that are children of BODY:

```
<body>
  <div>this is my Div
    <p>
      The style rule will not apply to this paragraph, coz it is not direct child
    </p>
  </div>
  <p>
      But will be applied to this paragraph
  </p>
</body>
```

Example!

Grouping

- Grouping selectors is done by separating each selector with a comma to give the same properties to a number of selectors without having to repeat

h1,h2,h3,h4,h5,h6 { color: green; font-family: "Arial" }

Selectors

Example:

h1 { font-family: “sans-serif “}
h2 { font-family: “sans-serif” }
h3 { font-family: “sans-serif “}

is equivalent to:

h1, h2, h3 { font-family: “sans-serif” }

CSS Rules Measurement Units

- Physical Measurements
  - inches (in)
  - points (pt)
- Screen Measurements
  - pixels (px)
- Relative Measurements
  - %
  - em
- Zero can be used with no unit

https://www.w3.org/Style/Examples/007/units.en.html

Colors Values

- Colors are set in RGB format represented as
  - hex representation
    - e.g. #FFCC99 ← #RRGGBB
  - Short hex representation
    - e.g. #FC9 ← #RGB
  - Predefined color aliases / keyword
    - e.g. black, red, blue, etc.
  - rgb(R, G, B [,A]) property
    - e.g. rgb(0,0,0) ➔ #000000 ➔ black

      rgb(255,255,255) ➔ #FFFFFF ➔ white

Styles Categories

1. ~~Box Model (Borders, Padding, and Margins)~~
2. ~~Font Styles~~
3. Text Styles
4. ~~Text and Background Colors~~
5. ~~Background Images~~
6. List Style

Font Style

CSS Property	Values
font-family:***name***	**Font *name* can be any system font; multiple names can be specified in order of preference, separated by commas.**
font-size:***size***	**Font *size* is specified as in a unit of measurement, normally point size (12pt).**
font-style:***style***	**Font *style* specified as** **normal** **italic**
font-weight:***weight***	**Font *weight* specified as** **normal** **bold**
font-variant:***variant***	**Font *variant* specified as** **normal** **small-caps**

Text Style

CSS Property	Values
text-align:***alignment***	**Sets the horizontal *alignment* of text within an element. The *alignment* can be:** **left** **center** **right** **justify**
line-height:***height***	**Sets the *height* of lines of text in an element; specify a measurement (px, pt, *n*%, em)** **normal**
letter-spacing:***spacing***	**Sets the *spacing* between letters in an element; specify a measurement (px, pt, *n*%, em)** **normal**
word-spacing:spacing	**Declares the space between words in the text.; specify a measurement (px, pt, *n*%, em)** **normal**

Text Style

CSS Property	Values
text-indent:***size***	Sets the *size* of indentation of the first line of a block of text; specify units of measurement (px, pt, *n*%, em)
text-transform:***case***	Sets the *case* of words in a text block using capitalize (First letter) lowercase uppercase (whole word) none
text-decoration:***style***	Sets a *style* using: underline overline line-through none

Text and Background Colors

CSS Property	Values
color:***color***	**Foreground color specified as a color name, hexadecimal value, or RGB value:** **color:red** **color:#FF0000** **color:rgb(255,0,0)**
background-color:***color***	**Background color specified as a color name, hexadecimal value, or RGB value:** **background-color:red** **background-color:#FF0000** **background-color:rgb(255,0,0)**

Example!

Background Images

CSS Property	Values
background-image:url(*url*)	Sets the URL of a background image; *url* can be set to none to prevent an image from loading.
background-position:*location*	Sets the *location* of the left and top edges of the background image with a pair of values separated by a space. Values are left|center|right paired with top|center|bottom
background-repeat:*axes*	Sets whether a background image should repeat along the horizontal and/or vertical axes. *Axes* values are: no-repeat repeat repeat-x repeat-y
background-attachment:*value*	Describes whether a background image remain fixed in place or scrolls with the document. *Values* are: fixed scroll

List Style

- list-style-type: circle | disc | square | armenian | decimal | decimal-leading-zero | georgian | lower-alpha | lower-greek | lower-latin | lower-roman | upper-alpha | upper-latin | upper-roman | none | *hebrew* | *armenian* | *hiragana* | *katakana* | *hiragana-iroha* | *katakana-iroha* ;
- list-style-position: inside | outside ;
- list-style-image: *uri* | none ;
- list-style: *list-style-type list-style-position list-style-image* ;

Concepts you should know

- ~~Grouping~~
- ~~Cascading~~
- Inheritance
- ! Important
- Specificity

Inheritance

- **Inheritance** is the process by which properties are passed down from parent to child elements even though those properties have not been explicitly defined by other means.

- Certain properties are inherited automatically, and as the name implies, a child element will take on the characteristics of its parent with regards to these properties.

Inheritance

- Some CSS styles are inherited and some not
  - Text-related and list-related properties are inherited
    - e.g. color, font-size, font-family, line-height, text-align, list-style, etc.
  - Box-related are not inherited
    - e.g. width, height, border, margin, padding, position, float, etc.
  - <a> elements do not inherit color and text-decoration
  - Color property is also inherited

Specificity

- **Specificity** is a mechanism within the CSS cascade that aids conflict resolution.
- It is used to determine the precedence of CSS style declarations with the same origin.

http://anaturb.net/dojo/my/cascade.htm

- Selectors are what matters.
- If same number of points then, Order matters.

Specificity

A B C D

- **D** → Count of Types ~~and Pseudo-elements~~ selectors
- **C** → Count of Classes, Attributes, ~~& Pseudo-classes~~ selectors
- **B** → Count of ID selectors
- **A** → Count of Inline Style

Applying Specificity

- For each ID value, apply 0,1,0,0 points
- For each class value, apply 0,0,1,0 points
- For each element reference, apply 0,0,0,1 point

Specificity

Selector		A	B	C	D
#warning,p.message	#warning	0	1	0	0
	p.message	0	0	1	1
p#warning		0	1	0	1
p.message		0	0	1	1
p		0	0	0	1

Specificity Important Notes

- The universal selector (*) has no specificity value
- If the element has inline styling, that automatically wins
- If two or more selectors have the same specificity, then, the latter specified rule takes precedence.
- The **!important** value appended a CSS property value is an *automatic win*.

CSS Online References

- http://reference.sitepoint.com/css
- http://www.tutorialspoint.com
- http://css-tricks.com/almanac/
- http://www.w3.org/community/webed/wiki/CSS/Properties
- https://developer.mozilla.org/en-US/docs/Web/CSS/Reference
- http://flexbox.help/
- https://css-tricks.com/snippets/css/a-guide-to-flexbox/
- https://css-tricks.com/almanac/properties/c/clear/

Assignments