CSS -- Cascading Style Sheet

Prepared by: Web Team

CSS Introduction

1. What is CSS?

- CSS stands for Cascading Style Sheets
- Styles define **how to display** HTML elements
- Styles were added to HTML 4.0 **to solve a problem**
- **External Style Sheets** can save a lot of work
- External Style Sheets are stored in **CSS files**

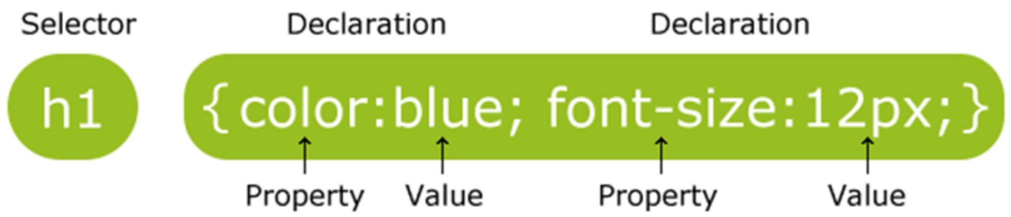
CSS Introduction (cont.)

2. CSS Saves a lot of Work!

   - CSS defines **HOW** HTML elements are displayed.
   - Styles are normally saved in external **.css** files. External style sheets enable you to change the appearance and layout of all the pages in a website, just by editing one single file!

CSS Syntax

1. CSS Syntax



- The selector is normally the HTML element you want to style.
- Each declaration consists of a property and a value.
- The property is the style attribute you want to change. Each property has a value.

CSS Syntax (cont.)

2. CSS Example

- A CSS declaration always
  - ends with a semicolon, and
  - declaration groups are surrounded by curly brackets:
  - **Example:**

```
p {color:red;text-align:center;}
```

CSS Syntax (cont.)

2. CSS Example (cont.)

- To make the CSS more readable, you can put one declaration on each line, like this:
  - **Example:**

```
p {
  color:red;
  text-align:center;
}
```

CSS Syntax (cont.)

3. CSS Comments
    - Comments are used to explain your code, and may help you when you edit the source code at a later date.
    - Comments are ignored by browsers.
    - A CSS comment begins with "/*", and ends with "*/", like this:

```
/*This is a comment*/
p {
    text-align:center;
    /*This is another comment*/
    color:black;
}
```

CSS How to

When a browser reads a style sheet, it will format the document according to it.

1. Three Ways to Insert CSS
   - There are three ways of inserting a style sheet:
     - External style sheet
     - Internal style sheet
     - Inline style

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
    - External style sheet
        - An external style sheet is ideal when the style is applied to many pages.
        - With an external style sheet, you can change the look of an entire Web site by changing one file.Each page must link to the style sheet using the <link> tag.

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
   - External style sheet (cont.)
     - The <link> tag goes inside the head section:

```
<head>
     <link rel="stylesheet" href="mystyle.css">
</head>
```

     - An external style sheet can be written in any text editor. The file should not contain any html tags. Your style sheet must be saved with a **.css** extension.

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
    - External style sheet (cont.)
        - An example of a style sheet file is shown below:

```
hr {border-color: sienna;}
p {margin-left: 20px;}
body {background-image:url("images/background.gif");}
```

***Note**: Do not add a space between the property value and the unit (such as margin-left:20 px). The correct way is: margin-left:20px.*

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
    - Internal Style Sheet
        - An internal style sheet may be used if one single HTML page has a unique style.
        - The internal style is defined inside the **<style>** element, inside the head section:

```
<style>
      * {
        background-color: yellowgreen;
      }
</style>
```

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
   - Inline Styles
     - An inline style loses many of the advantages of style sheets by mixing content with presentation. Use this method sparingly!
     - To use inline styles you use the style attribute in the relevant tag. The style attribute can contain any CSS property.

CSS How to (cont.)

1. Three Ways to Insert CSS (cont.)
   - Inline Styles (cont.)
     - The example shows how to change the color and the left margin of a paragraph:

```
<p style="color:sienna;margin-left:20px;">
    This is a paragraph.
</p>
```

CSS How to (cont.)

2. Multiple Style Sheets

- If some **properties** have been **set** for the **same selector** in different style sheets, the **values** will be inherited from the more specific style sheet.

CSS How to (cont.)

3. Multiple Style Sheets - Example

- An **external style sheet** has these properties for the h3 selector:

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

CSS How to (cont.)

3. Multiple Style Sheets - Example (cont.)

- And an **internal style sheet** has these properties for the h3 selector:

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

CSS How to (cont.)

3. Multiple Style Sheets - Example (cont.)

- If the page with the **internal style sheet** also **links** to the **external style sheet** the **properties** for **h3** will be:

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

  - The **`color`** is inherited from the external style sheet and
  - the **`text-align`** and the **`font-size`** is replaced by the internal style sheet.

CSS How to (cont.)

3. <u>Multiple Styles Will Cascade into One</u>

- Styles can be specified:
  - inside an HTML element
  - inside the head section of an HTML page
  - in an external CSS file

***Tip**: Even multiple external style sheets can be referenced inside a single HTML document.*

CSS How to (cont.)

4. Cascading order
   - What style will be used when there is more than one style specified for an HTML element?
   - Generally speaking, we can say that all the styles will "cascade" into a new "virtual" stylesheet by the following rules, where number 4 has the highest priority:

     i. Browser default
     ii. External style sheet
     iii. Internal style sheet (in the head section)
     iv. Inline style (inside an HTML element)

CSS How to (cont.)

4. Cascading order (cont.)
   - So, an inline style (inside an HTML element) has the highest priority, which means that it will override a style defined inside the **<head>** tag, or in an external style sheet, or in a browser (a default value).

***Note**: If the link to the external style sheet is placed after the internal style sheet in HTML <head>, the external style sheet will override the internal style sheet!*

CSS Id & Class

In addition to setting a style for a HTML element, CSS allows you to specify your own selectors called "id" and "class".

1. The id Selector
   - The id selector is used to specify a style for a single, unique element.
   - The id selector uses the id attribute of the HTML element, and is defined with a "#".

CSS Id & Class (cont.)

1. The id Selector (cont.)

- The style rule below will be applied to the element with id="para1":

```
#para1 {
  text-align:center;
  color:red;
}
```

Note: Do NOT start an ID name with a number!

CSS Id & Class (cont.)

2. The class Selector

- The class selector is used to specify a style for a group of elements. Unlike the id selector, the class selector is most often used on several elements.
- This allows you to set a particular style for many HTML elements with the same class.

CSS Id & Class (cont.)

2. The class Selector (cont.)

- The class selector uses the HTML class attribute, and is defined with a "."
- In the example below, all HTML elements with class="center" will be center-aligned:

```
.center {
  text-align:center;
}
```

CSS Id & Class (cont.)

2. The class Selector (cont.)
   - You can also specify that only specific HTML elements should be affected by a class.
   - In the example below, **all p elements** with class="center" will be center-aligned:

```
p.center {
  text-align:center;
}
```

Note: Do NOT start a class name with a number!

CSS Selectors

In CSS, selectors are patterns used to select the element(s) you want to style. There are many selector patterns in CSS.

1. <u>* Selector</u> : selects all the elements

- Syntax:

```
* {
        css declaration;
}
```

- Example:

```
* {
   background-color : yellow;
}
```

CSS Selectors (cont.)

2. Element : select all the p elements

- Syntax:

```
element {
        css declaration;
}
```

- Example:

```
p {
  background-color : yellow;
}
```

CSS Selectors (cont.)

3. **Grouping Element** : is used to minimize the code when there are elements with the same style. You have to separate each selector with a comma.

- Syntax:

  element, element {

  css declaration

  }

- Example:

```
h1, h2 {
  background-color : yellow;
}
```

CSS Selectors (cont.)

4. Nesting Selectors :
    - It is possible to apply a style for a selector within a selector.
    - There are **2** ways to apply nesting selector.
        - Separate each other with a space:

          element element
        - Stand next to each other:

          element.class or element#id

CSS Selectors (cont.)

4. Nesting Selectors (cont.)
   - *element element*
     - Stand next to each other (space = nested)
     - Example:

```
div h2 {
  background-color : yellow;
}
```

CSS Selectors (cont.)

4. Nesting Selectors (cont.)
   - *element.class or element#id*
     - Separate each other with a space

       (without space = the same level)
     - Example:

```
div.red {
  background-color : red;
}
```

CSS Attribute Selectors

- The ***[attribute]*** selector is used to select elements with the specified attribute.
- Syntax:

```
[attribute] {
        css declarations;
}
```

CSS Attribute Selectors

- *There are*

  a. *[attribute] Selector*

  b. *[attribute = “value”] Selector*

  c. *[attribute ~= “value”] Selector*

  d. *[attribute |= “value”] Selector*

  e. *[attribute ^= “value”] Selector*

  f. *[attribute $= “value”] Selector*

CSS Attribute Selectors (cont.)

a. CSS [attribute] Selector

- Example:

```
a [target] {
    background-color : yellow;
}
```

CSS Attribute Selectors (cont.)

a. CSS [attribute = “value”] Selector

- is used to select elements with a specified attribute and value.
- Example:

```
a[target=“_blank”] {
    background-color : yellow;
}
```

CSS Attribute Selectors (cont.)

c. CSS [attribute ~= “value”] Selector

- is used to select elements with an attribute value containing a specified word
- Example:

```
    [title ~= “flower”] {
      border : 5px solid yellow;
    }
<img src="images/home.jpg" title="my flower is red"/>
*Case Sensitive
```

CSS Attribute Selectors (cont.)

d. CSS [attribute |= “value”] Selector

- is used to select elements with the specified attribute starting with the specified value.
- Example:

```
[class |= “top” ] {background : yellow; }
```

Note: The value has to be a whole word, either alone, like class="top", or followed by a hyphen(-), like class="top-text"! *First in hyphen word

CSS Attribute Selectors (cont.)

e. CSS [attribute ^= “value”] Selector

- is used to select elements whose attribute value begins with a specified value.
- Example:

```
[class ^= “top”] { background : yellow; }
```

***Note:** The value does not have to be a whole word!*

*First character set in word

CSS Attribute Selectors (cont.)

f. CSS [attribute $= "value"] Selector

- is used to select elements whose attribute value ends with a specified value.
- Example:

```
[class $= "top"] { background : yellow; }
```

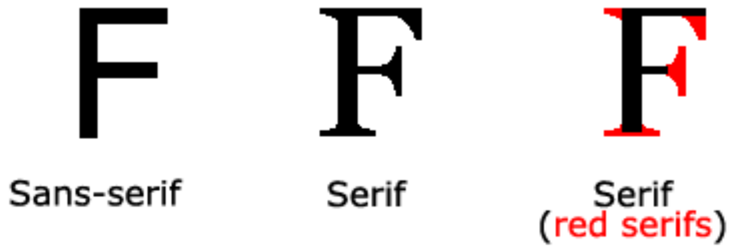
Note: The value does not have to be a whole word!

*last character set in word

CSS Font

- CSS font properties define the font family, boldness, size, and the style of a text.

1. Difference Between Serif and Sans-serif Typefaces



CSS Font (cont.)

2. CSS Font Families

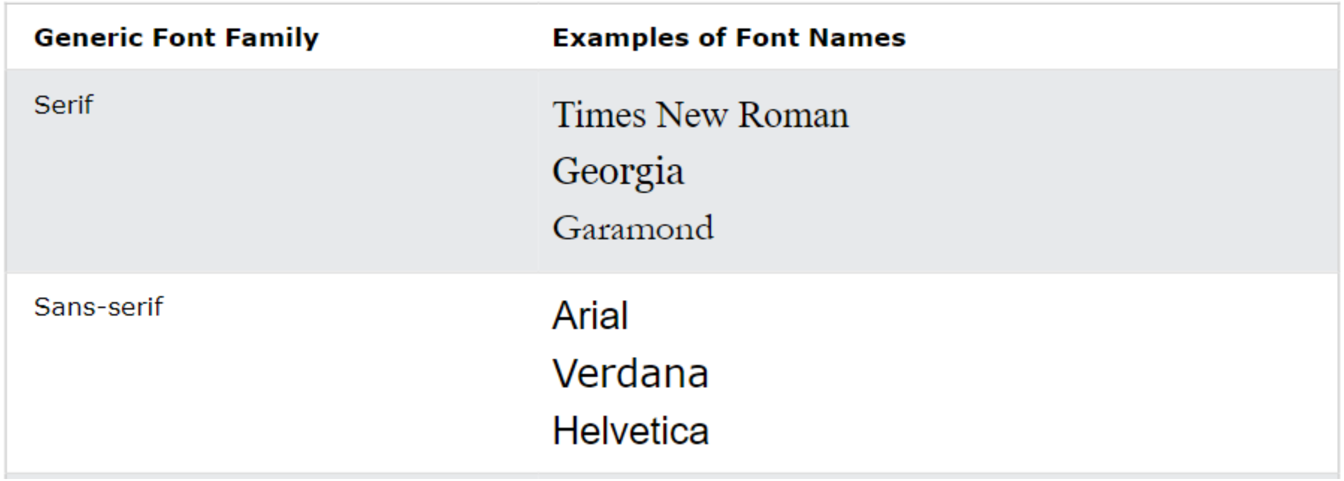
- In CSS, there are **two types** of font family names:
    - *Generic font families* - a group of font families with a similar look (like "Serif" or "Monospace")
    - *Specific font families* - a specific font family (like "Times New Roman" or "Arial")

***Note**: On computer screens, sans-serif fonts are considered easier to read than serif fonts.*

CSS Font (cont.)

2. CSS Font Families

- Two **types** of font family names:



Generic Font Family	Examples of Font Names
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica

CSS Font (cont.)

3. Font Family

- The font family of a text is set with the font-family property.
- The font-family property should hold several font names as a "**fallback**" system. If the browser does not support the first font, it tries the next font.
- Start with the font you want, and end with a **generic family**, to let the browser pick a similar font in the generic family, if no other fonts are available.

CSS Font (cont.)

3. Font Family (cont.)

- More than one font family is specified in a comma-separated
- Example:

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

***Note**: If the name of a font family is more than one word, it must be in quotation marks, like: "Times New Roman".*

CSS Font (cont.)

4. Font Family - Download Embed
   - In case, We want to use to support all browsers; we have to link or download that fonts to embed with your project.
   - There are
     - Link Google fonts
     - Download font

CSS Font (cont.)

4. Font Family - Download Embed (cont.)
   - Link Google fonts
     - **go to:** fonts.google.com
     - **search** name of fonts you love
     - **select** that fonts to get link.

CSS Font (cont.)

4. Font Family - Download Embed (cont.)

   - Link Google fonts (cont.)

     - To embed a font, copy the code into the **<head>** of your html:

```
<link href="https:..." rel="stylesheet">
```

     - CSS rules to specify families :

```
font-family: 'Roboto', sans-serif;
```

CSS Font (cont.)

4. Font Family - Download Embed (cont.)
   - Download font
     - download font file type: .ttf or .wotf, ...
     - copy that file to your project
     - write the code into the **<style>** element or css file

```
@font-face {
    font-family: "Fasthand-Regular";
    src: url("../fonts/Fasthand-Regular.ttf");
}
```

CSS Font (cont.)

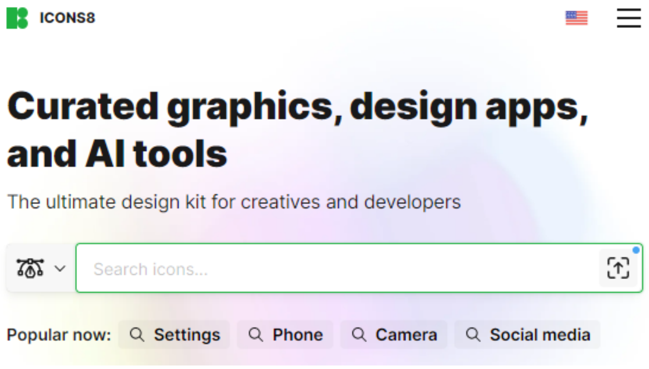
5. Icon

- Icon: Small pictorial symbol used in a graphical user interface (GUI) or in web documents to identify a file, folder, program, or device (such as drive, modem, printer).
- There are a lot of website that publish icon with different ui such as fontawesome, icon8, flaticon, googleicon……….
- All of icons above is required to link to css file or download it.

CSS Font (cont.)

5. Icon (cont.)

- How to use Icon8
- How to use font awesome



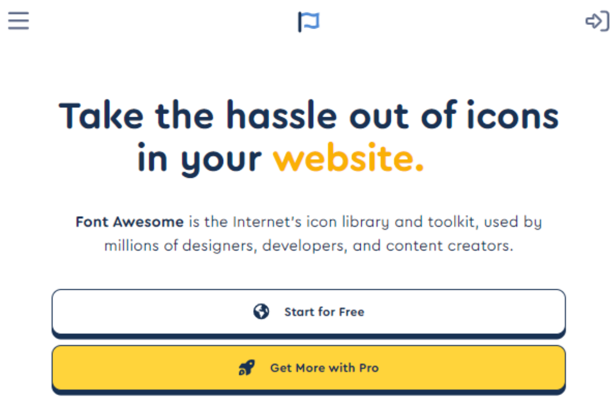
ICONS8

Curated graphics, design apps, and AI tools

The ultimate design kit for creatives and developers

Search icons...

Popular now: Settings Phone Camera Social media



Take the hassle out of icons in your website.

Font Awesome is the Internet's icon library and toolkit, used by millions of designers, developers, and content creators.

Start for Free

Get More with Pro

CSS Font (cont.)

6. Font Size

- The **font-size** property sets the size of the text.
- Being able to manage the text size is important in web design. However, you should not use font size adjustments to make paragraphs look like headings, or headings look like paragraphs.
- Always use the proper HTML tags, like <h1> - <h6> for headings and <p> for paragraphs.

CSS Font (cont.)

6. Font Size (cont.)

- Example

```
h1 {font-size:2.5em;} /* 40px/16=2.5em */
h2 {font-size:1.875em;} /* 30px/16=1.875em */
p {font-size:0.875em;} /* 14px/16=0.875em */
```

- In the example above, the text size in **em** is the same as the previous example in pixels. However, with the **em** size, it is possible to adjust the text size in all browsers.

CSS Font (cont.)

7. Use a Combination of Percent and Em

- The solution that works in all browsers, is to set a default font-size in percent for the <body> element:
- Example

```
body {font-size:100%;}
h1 {font-size:2.5em; p {font-size:0.875em;}
```

- Our code now works great! It shows the same text size in all browsers, and allows all browsers to zoom or resize the text!

CSS Link

Links can be styled in different ways.

1. Styling Links

   - Links can be styled with any CSS property (e.g. color, font-family, background, etc.).
   - In addition, links can be styled differently depending on what **state** they are in.

CSS Link (cont.)

1. Styling Links (cont.)
   - The four links states are:
     - *a:link* - a normal, unvisited link
     - *a:visited* - a link the user has visited
     - *a:hover* - a link when the user mouses over it
     - *a:active* - a link the moment it is clicked

CSS Link (cont.)

1. Styling Links (cont.)

- Example

```
a:link {color:#FF0000;}       /* unvisited link */
a:visited {color:#00FF00;}  /* visited link */
a:hover {color:#FF00FF;}  /* mouse over link */
a:active {color:#0000FF;}  /* selected link */
```

CSS Link (cont.)

1. Styling Links (cont.)

   - When setting the style for several link states, there are some order rules :
     - *a:link*
     - *a:visited*
     - *a:hover*
     - *a:active*

CSS Link (cont.)

2. Common Link Styles

- In the example the link changes color depending on what state it is in.
- Example

```
a:link {color:#FF0000;}       /* unvisited link */
a:visited {color:#00FF00;}  /* visited link */
a:hover {color:#FF00FF;}  /* mouse over link */
a:active {color:#0000FF;}  /* selected link */
```

CSS Link (cont.)

3. Text Decoration

- The **text-decoration** property is mostly used to remove underlines from links:

- Example

```
a:link {text-decoration:none;}
a:hover {text-decoration:underline;}
a:active {text-decoration:underline;}
```

CSS Link (cont.)

4. Background Color

- The **background-color** property specifies the background color for links:
- Example

```
a:link {background-color:#B2FF99;}
a:visited {background-color:#FFFF85;}
a:hover {background-color:#FF704D;}
a:active {background-color:#FF704D;}
```

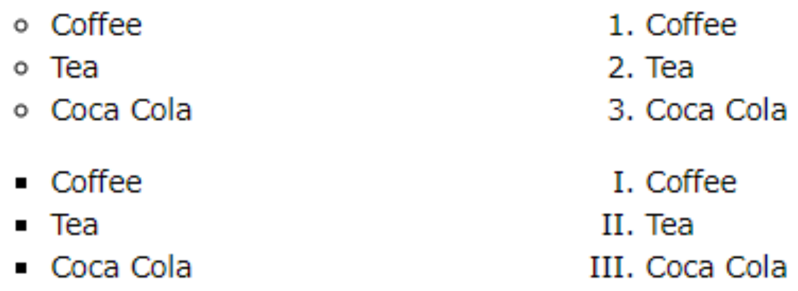
CSS List (cont.)

1. List

- In HTML, there are **two** types of lists:
    - unordered lists - the list items are marked with bullets
    - ordered lists - the list items are marked with numbers or letters
- With CSS, lists can be styled further, and images can be used as the list item marker.

CSS List

- The CSS **list** properties allow you to:
    - Set different list item markers for ordered lists
    - Set different list item markers for unordered lists
    - Set an image as the list item marker



```
∘ Coffee            1. Coffee
∘ Tea               2. Tea
∘ Coca Cola         3. Coca Cola

▪ Coffee            I. Coffee
▪ Tea              II. Tea
▪ Coca Cola       III. Coca Cola
```

CSS List (cont.)

2. Different List Item Markers

- The type of list item marker is specified with the list-style-type property.

```
ul.a {list-style-type: circle;}

ul.b {list-style-type: square;}

ol.c {list-style-type: upper-roman;}

ol.d {list-style-type: lower-alpha;}
```

- Some of the values are for unordered lists, and some for ordered lists.

CSS List (cont.)

2. An Image as The List Item Marker

- To specify an image as the list item marker, use the list-style-image property.
- Example:

```
ul {
    list-style-image: url('sqpurple.gif');
}
```

CSS List (cont.)

2. An Image as The List Item Marker (cont.)

    - The example does not display equally in all browsers. IE and Opera will display the image-marker a little bit higher than Firefox, Chrome, and Safari.

CSS List (cont.)

4. List - Shorthand property

- It is also possible to specify all the list properties in one, single property. This is called a shorthand property.
- The shorthand property used for lists, is the list-style property:
- Example

```
ul {
    list-style: square outside url("sqpurple.gif");
}
```

CSS List (cont.)

4. <u>List - Shorthand property (cont.)</u>

- When using the shorthand property, the order of the values are:
  - *list-style-type*
  - *list-style-position*
  - *list-style-image*
- It does not matter if one of the values above are missing, as long as the rest are in the specified order.

CSS Table

1. Table Borders

- To specify table borders in CSS, use the border property.
- The example below specifies a black border for table, th, and td elements
- Example

```
table, th, td {
    border: 1px solid black;
}
```

CSS Table (cont.)

1. Table Borders
   - Notice that the table in the example has double borders. This is because both the table and the th/td elements have separate borders.
   - To display a single border for the table, use the border-collapse property.

CSS Table (cont.)

2. Collapse Borders

- The **border-collapse** property sets whether the table borders are collapsed into a single border or separated:
- Example

```
table { border-collapse:collapse; }
table, th, td { border: 1px solid black; }
```

CSS Table (cont.)

3. **Table Width and Height**

- Width and height of a table is defined by the width and height properties.
- The example below sets the width of the table to 100%, and the height of the th elements to 50px:
- Example

```
table { width:100%; }        th { height:50px; }
```

CSS Table (cont.)

4. Table Text Alignment
   - The text in a table is aligned with
     - the **text-align** properties,
     - and **vertical-align** properties.

CSS Table (cont.)

4. Table Text Alignment (cont.)
   - The **text-align** properties
     - sets the horizontal alignment, like left, right, or center:
     - Example

```
td {
    text-align:right;
}
```

CSS Table (cont.)

4. Table Text Alignment (cont.)
    - The **vertical-align** properties
        - sets the vertical alignment, like top, bottom, or middle:
        - Example

```
td  {
    height:50px;
    vertical-align:bottom;
}
```

CSS Table (cont.)

5. Table Padding

- To control the space between the border and content in a table, use the padding property on td and th elements:
- Example

```
td {
     padding:15px;
}
```

CSS Table (cont.)

6. Table Color

- The example below specifies the color of the borders, and the text and background color of th elements:
- Example

```
table, td, th {
    border:1px solid green;
}
th {
    background-color:green;
    color:white;
}
```

CSS Background

- CSS **background** properties are used to define the background effects of an element.
- CSS properties used for background effects:
    - background-color
    - background-image
    - background-repeat
    - background-attachment
    - background-position
    - background-size
    - background-origin

CSS Background (cont.)

1. background-color

- The background-color property specifies the background color of an element.
- The background color of a page is defined in the body selector:
- **Example:**

```
body {
    background-color: #b0c4de;
}
```

CSS Background (cont.)

1. background-color (cont.)

- With CSS, a color is most often specified by:
  - a HEX value - like "#ff0000"
  - an RGB value - like "rgb(255,0,0)“ and “rbga(0,255,255,0.4)”
  - a color name - like "red"

CSS Background (cont.)

1. background-color (cont.)

- **Example:** h1, p, and div elements have different background colors:

```
h1  { background-color:#6495ed; }

p   { background-color:#e0ffff; }

div { background-color:#b0c4de; }
```

CSS Background (cont.)

2. background-image

- The background-image property specifies an image to use as the background of an element.
- By default, the image is repeated so it covers the entire element.

CSS Background (cont.)

2. background-image

- **Example:** The background image for a page can be set like this:

```
body { background-image:url("paper.gif"); }
```

- **Example:** Below is an example of a bad combination of text and background image. The text is almost not readable:

```
body { background-image:url("bgdesert.jpg"); }
```

CSS Background (cont.)

3. Background Image - Repeat Horizontally or Vertically
   - By default, the background-image property repeats an image both horizontally and vertically.
   - Some images should be repeated only horizontally or vertically, or they will look strange, like this:
   - **Example:**

```
body {
  background-image:url("gradient2.png");
}
```

CSS Background (cont.)

3. Background Image - Repeat Horizontally or Vertically (cont.)

- If the image is repeated only horizontally (repeat-x), the background will look better:
- **Example:**

```
body {
  background-image:url("gradient2.png");
  background-repeat:repeat-x;
}
```

CSS Background (cont.)

4. **Background Image - Set position and no-repeat**

- Showing the image only once is specified by the background-repeat property:
    - **Example:**

```
body {
        background-image:url("img_tree.png");
        background-repeat:no-repeat;
 }
```

CSS Background (cont.)

4. Background Image - Set position and no-repeat

    - In the example above, the background image is shown in the same place as the text. We want to change the position of the image, so that it does not disturb the text too much.

CSS Background (cont.)

4. Background Image - Set position and no-repeat (cont.)

- The position of the image is specified by the background-position property:
  - **Example:**

```
body {
      background-image:url("img_tree.png");
      background-repeat:no-repeat;
      background-position:right top;
    }
```

CSS Background (cont.)

5. Background - Shorthand property

- As you can see from the examples above, there are many properties to consider when dealing with backgrounds.
- To shorten the code, it is also possible to specify all the properties in one single property. This is called a shorthand property.

CSS Background (cont.)

5. Background - Shorthand property
    - The shorthand property for background is simply "background":
    - Example

```
body {
  background: #ffffff url("img_tree.png") no-repeat
right top;
}
```

CSS Background (cont.)

5. Background - Shorthand property (cont.)

- When using the shorthand property the order of the property values is:
  - background-color
  - background-image
  - background-repeat

CSS Background (cont.)

5. Background - Shorthand property (cont.)
    - background-attachment
    - background-position
  - It does not matter if one of the property values is missing, as long as the ones that are present are in this order.

CSS Background (cont.)

6. CSS3 Multiple Background

- CSS3 allows you to add multiple background images for an element, through the background-image property
- The different background images are separated by commas, and the images are stacked on top of each other, where the first image is closest to the viewer.

CSS Background (cont.)

6. CSS3 Multiple Background (cont.)

- Example:

```
#example1 {
  background-image: url(img_flwr.gif),
url(paper.gif);
  background-position: right bottom, left top;
  background-repeat: no-repeat, repeat;
}
```

CSS Background (cont.)

6. CSS3 Multiple Background (cont.)
    - The example above has two background images, the 1st image is a flower (aligned to the bottom and right) & the 2nd image is paper background (aligned to the top-left corner)

CSS Background (cont.)

6. CSS3 Multiple Background (cont.)

- Multiple background images can be specified using either the individual background properties (as above) or the background shorthand property.

```
#example1 {
        background: url(img_flwr.gif) right bottom no-
      repeat,
                    url(paper.gif) left top repeat;
                  }
```

CSS Background (cont.)

7. CSS3 Background Size

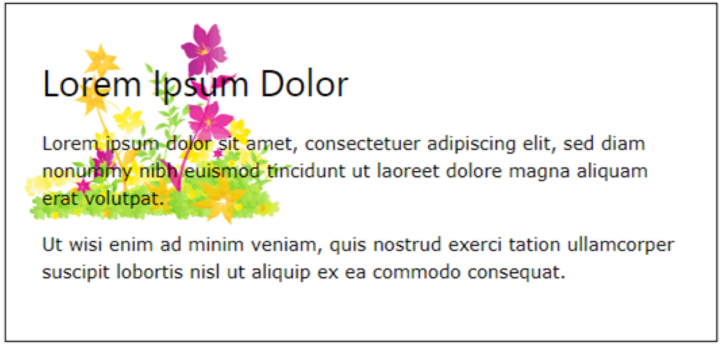
- The CSS3 *background-size* property allows you to specify the size of background images.
- Before CSS3, the size of a background image was the actual size of the image. CSS3 allows us to reuse background images in different contexts.
- The size can be specified in lengths, percentages, or by using one of the two keywords: *contain* or *cover*.

CSS Background (cont.)

7. CSS3 Background Size (cont.)

- The following example resizes a background image to much smaller than the original image (using pixels):

Original background image:

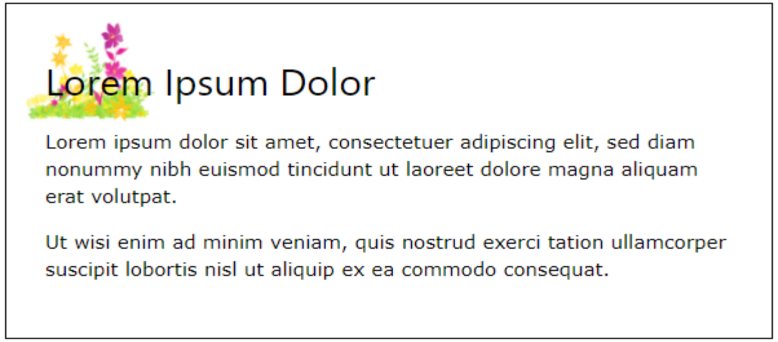


Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Resized background image:



Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

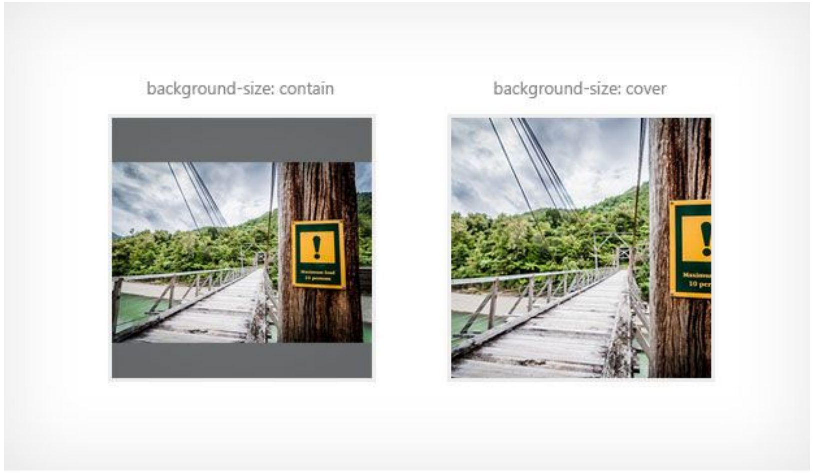
CSS Background (cont.)

7. CSS3 Background Size (cont.)

- Example:
```
#div1 {
    background: url(img_flower.jpg);
    background-size: 100px 80px;
    background-repeat: no-repeat;
}
```
- The two other possible values for background-size are
    - **contain** and **cover**

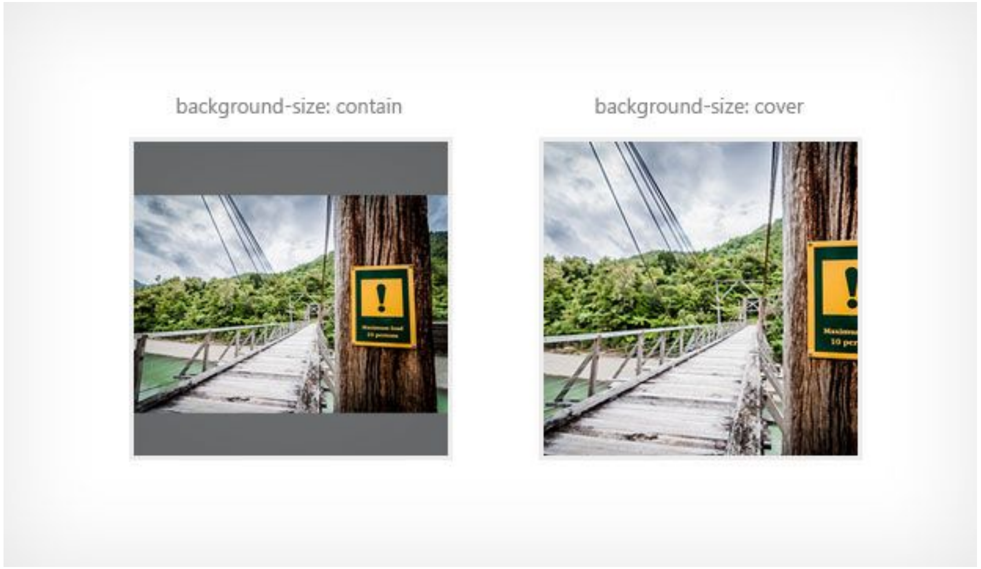
CSS Background (cont.)

- **The contain keyword**:
    - scales the background image to be as large as possible (but both its width and its height must fit inside the content area).



CSS Background (cont.)

- As such, depending on the proportions of the background image and the background positioning area, there may be some areas of the background which are not covered by the background image.



CSS Background (cont.)

- The cover keyword:
  - *scales the background image so that the content area is completely covered by the background image (both its width and height are equal to or exceed the content area)*. As such, some parts of the background image may not be visible in the background positioning area.

CSS Background (cont.)

7. CSS3 Background Size (cont.)

- Example: This example illustrates the use of *contain* and *cover.*

```
#div1 {
  background: url(img_flower.jpg);
  background-size: contain;
  background-repeat: no-repeat;
}

#div2 {
  background: url(img_flower.jpg);
  background-size: cover;
  background-repeat: no-repeat;
}
```

CSS Background (cont.)

8. **Define Sizes of Multiple Background Images**

- The ***background-size*** property also accepts multiple values for background size (using a comma-separated list), when working with multiple backgrounds.

CSS Background (cont.)

8. Define Sizes of Multiple Background Images

- The following example has three background images specified, with different background-size value for each image:

```
#example1 { background: url(img_flwr.gif) left top no-repeat,
                     url(img_flwr.gif) right bottom no-
repeat,
                     url(paper.gif) left top repeat;
          background-size: 50px, 130px, auto; }
```

CSS Background (cont.)

9. Full Size Background Image

- Now we want to have a background image on a website that covers the entire browser window at all times.
- The requirements are as follows:
  - Fill the entire page with the image (no white space)
  - Scale image as needed
  - Center image on page
  - Do not cause scrollbars

CSS Background (cont.)

9. Full Size Background Image (cont.)

- The following example shows how to do it; Use the html element (the html element is always at least the height of the browser window). Then set a fixed and centered background on it. Then adjust its size with the background-size property:
- Example:

```
html {
  background: url(img_flower.jpg) no-repeat center fixed;
  background-size: cover;
}
```

CSS Background (cont.)

10. CSS3 Background-origin Property

- The CSS3 background-origin property specifies where the background image is positioned.
- The property takes three different values:
  - ***border-box*** - the background image starts from the upper left corner of the border

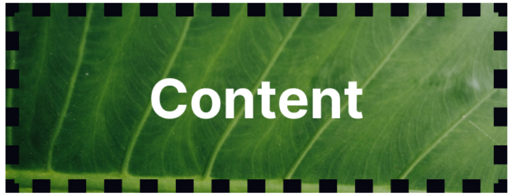
CSS Background (cont.)

10. <u>CSS3 Background-origin Property</u>

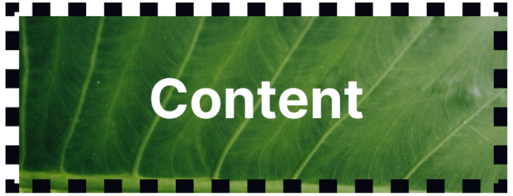
  - *padding-box* - (default) the background image starts from the upper left corner of the padding edge
  - *content-box* - the background image starts from the upper left corner of the content

CSS Background (cont.)

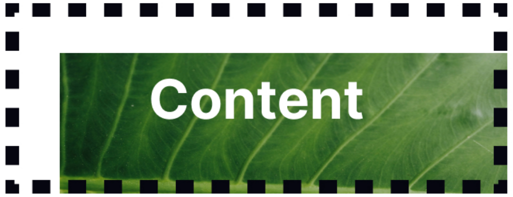
10. CSS3 Background-origin Property



```
background-origin:
border-box;
```



```
background-origin:
padding-box;
```



```
background-origin:
content-box;
```

CSS Background (cont.)

10. CSS3 Background-origin Property (cont.)

- **Example:**

```
#example1 {
  border: 10px solid black;
  padding: 35px;
  background: url(img_flwr.gif);
  background-repeat: no-repeat;
  background-origin: content-box;
}
```

***Note**: This property has no effect if background-attachment is "fixed".*

CSS Text

1. Text Color

- The color property is used to set the color of the text.
- The default color for a page is defined in the body selector.
- Example

```
body {color:blue;}
h1 {color:#00ff00;}
h2 {color:rgb(255,0,0);}
```

CSS Text (cont.)

2. <u>Text Alignment</u>

- The **text-align** property is used to set the horizontal alignment of a text.
- Text can be
  - **centered**,
  - or aligned to the **left**
  - or **right**,
  - or **justified**.

CSS Text (cont.)

2. Text Alignment (cont.)

- When **text-align** is set to "**justify**",
  - each line is stretched so that every line has equal width,
  - and the left and right margins are straight (like in magazines and newspapers).

- Example

```
h1 {text-align:center;}
p.date {text-align:right;}
p.main {text-align:justify;}
```

CSS Text (cont.)

3. Text Decoration

- The **text-decoration** property is used to set or remove decorations from text.
- It is used to remove underlines from links for design purposes.
  - Example `a {text-decoration:none;}`
- It can also be used to decorate text
  - Example

```
h1 {text-decoration:overline;}
h2 {text-decoration:line-through;}
h3 {text-decoration:underline;}
```

CSS Text (cont.)

4. Text Transformation

- The text-transform property is used to specify uppercase and lowercase letters in a text.
- It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word.
- Example

```
p.uppercase  { text-transform:uppercase;  }
p.lowercase  { text-transform:lowercase;  }
p.capitalize { text-transform:capitalize; }
```

CSS Text (cont.)

5. Text Indentation

- The text-indent property is used to specify the indentation of the first line of a text.
- Example

```css
p { text-indent: 50px; }
```

6. Letter Spacing

- The letter-spacing property is used to specify the space between the characters in a text.
- Example:

```css
h1 { letter-spacing: 3px;  }
h1 { letter-spacing: -3px; }
```

CSS Text (cont.)

7. Line Height

- The line-height property is used to specify the space between lines.
- Example `p.small { line-height: 0.8; }`

8. Text Direction

- The direction property is used to change the text direction.
- Example: `p { direction: rtl; } ( rtl = right to left )`

CSS Text (cont.)

9. Word Spacing

- The word-spacing property is used to specify the space between the words in a text.
- Example

```
h1 { word-spacing: 10px; }
```

CSS Text (cont.)

9. Text Shadow

- The **text-shadow** property adds shadows to text.
- **Syntax :** `text-shadow: h-shadow v-shadow blur-radius color;`
- **Example:**

```
h1 { text-shadow: 3px 2px 5px red; }
```

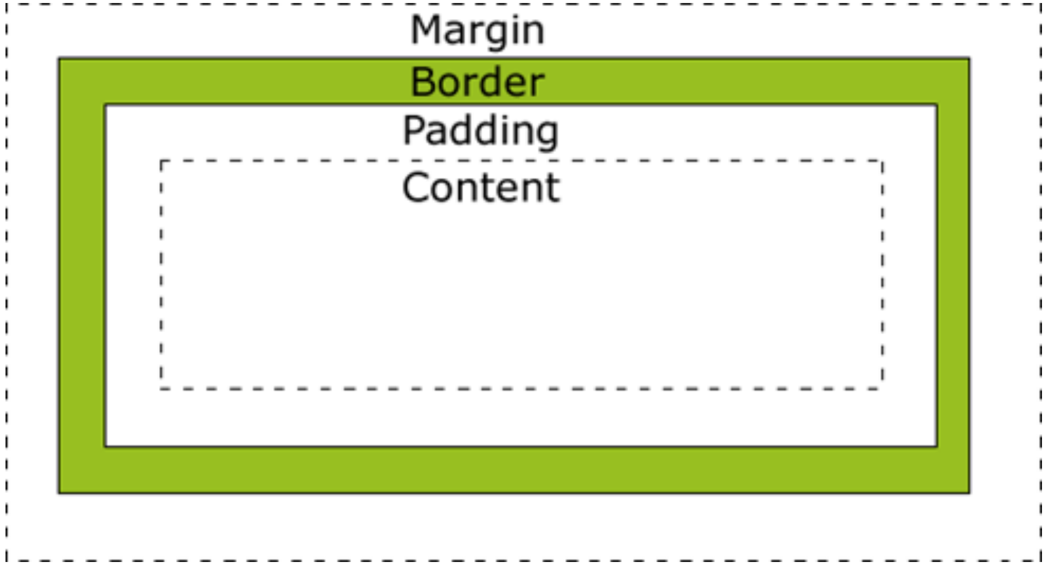
CSS Box Model

1. The CSS Box Model

- All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.
- The CSS box model is essentially a box that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content.
- The box model allows us to place a border around elements and space elements in relation to other elements.

CSS Box Model (cont.)

1. The CSS Box Model (cont.)
   - Explanation of the different parts:



CSS Box Model (cont.)

1. The CSS Box Model (cont.)

- Explanation of the different parts (cont.)
  - **Margin** - Clears an area around the border. The margin does not have a background color, it is completely transparents
  - **Border** - A border that goes around the padding and content. The border is inherited from the color property of the box.

CSS Box Model (cont.)

1. The CSS Box Model (cont.)
   - Explanation of the different parts (cont.)
     - **Padding** - Clears an area around the content. The padding is affected by the background color of the box
     - **Content** - The content of the box, where text and images appear.

CSS Box Model (cont.)

2. Width and Height of an Element
   - **Important:** When you set the **width** and **height** properties of an element with CSS,
     - you just set the **width** and **height** of the content area.
     - To calculate the **full size** of an element, you must also add
       - the padding,
       - borders
       - and margins.

CSS Box Model (cont.)

2. Width and Height of an Element (cont.)

- The total width of the element in the example below is:
- **Example:**

```
width:250px;
padding:10px;
border:5px solid gray;
margin:10px;
```

Let's do the math:
250px (width)
+ 20px (left + right padding)
+ 10px (left + right border)
+ 20px (left + right margin)
= **300**px

CSS Box Model (cont.)

2. **Width and Height of an Element (cont.)**

- Assume that you had only 250px of space. Let's make an element with a total width of 250px:
- **Example:**

```
width:220px;
padding:10px;
border:5px solid gray;
margin:0px;
```

CSS Box Model (cont.)

2. Width and Height of an Element (cont.)

- The total width of an element should be calculated like this:
  - **Total element width = width + left padding + right padding + left border + right border + left margin + right margin**

CSS Box Model (cont.)

2. Width and Height of an Element (cont.)
   - The total height of an element should be calculated like this:
     - **Total element height = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin**

CSS Borders

1. CSS Border Properties

   - The CSS border properties allow you to specify the style and color of an element's border.

2. Border Style

   - The border-style property specifies what kind of border to display.

     ***Note**: None of the border properties will have ANY effect unless the border-style property is set!*

CSS Borders (cont.)

2. Border Style (cont.)
   - border-style values:
     - none
     - dotted
     - dashed
     - solid
     - double
     - groove
     - ridge
     - inset
     - outset

CSS Borders (cont.)

3. Border Width

- The border-width property is used to set the width of the border.
- The width is set in pixels, or by using one of the three predefined values: thin, medium, or thick.
- Example

```
p.one {
    border-style:solid;
    border-width:5px;
}
```

```
p.two {
    border-style:solid;
    border-width:medium;
}
```

***Note**: The "border-width" property does not work if it is used alone. Use the "border-style" property to set the borders first.*

CSS Borders (cont.)

4. Border Color

- The border-color property is used to set the color of the border.
- You can also set the border color to "transparent".
- Example

```
p.one {
    border-style:solid;
    border-color:red;
}
```

```
p.two {
    border-style:solid;
    border-color:#98bf21;
}
```

***Note**: The "border-color" property does not work if it is used alone. Use the "border-style" property to set the borders first.*

CSS Borders (cont.)

5. Border - Individual sides

- In CSS it is possible to specify different borders for different sides:
  - Example

```
p { border-top-style:dotted;
    border-right-style:solid;
    border-bottom-style:dotted;
    border-left-style:solid;
    }
```

- The example above can also be set with a single property:
  - Example

```
border-style:dotted solid;
```

CSS Borders (cont.)

5. Border - Individual sides (cont.)

- The border-style property can have from **one to four** values.

   a. four values

      - `border-style:dotted solid double dashed;`
         - top border is dotted
         - right border is solid
         - bottom border is double
         - left border is dashed

CSS Borders (cont.)

5. Border - Individual sides (cont.)
   b. three values
      - `border-style:dotted solid double;`
        - top border is dotted
        - right and left borders are solid
        - bottom border is double

CSS Borders (cont.)

5. Border - Individual sides (cont.)

   c. two values

      - `border-style:dotted solid;`
         - top and bottom borders are dotted
         - right and left borders are solid

   d. one value

      - `border-style:dotted;`
         - all four borders are dotted

CSS Borders (cont.)

6. Border - Shorthand property
    - As you can see from the examples above, there are many properties to consider when dealing with borders.
    - To shorten the code, it is also possible to specify all the individual border properties in one property. This is called a shorthand property.

CSS Borders (cont.)

6. <u>Border - Shorthand property (cont.)</u>

- The border property is a shorthand for the following individual border properties:
    - border-width
    - border-style (required)
    - border-color
- Example `border:5px solid red;`

CSS Margin

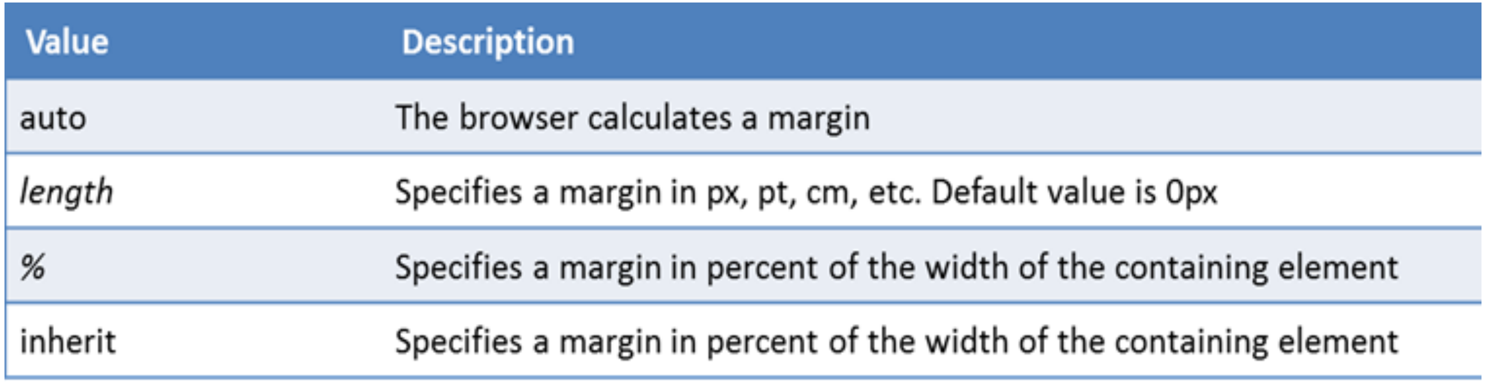
The CSS margin properties define the space around elements.

1. Margin
   - The margin clears an area around an element (outside the border). The margin does not have a background color, and is completely transparent.
   - The top, right, bottom, and left margin can be changed independently using separate properties. A shorthand margin property can also be used, to change all margins at once.

CSS Margin (cont.)

1. <u>Margin (cont.)</u>

   - Possible Values



Value	Description
auto	The browser calculates a margin
length	Specifies a margin in px, pt, cm, etc. Default value is 0px
%	Specifies a margin in percent of the width of the containing element
inherit	Specifies a margin in percent of the width of the containing element

CSS Margin (cont.)

2. Margin - Individual sides

- In CSS, it is possible to specify different margins for different sides:
- Example

```
margin-top:100px;
margin-bottom:100px;
margin-right:50px;
margin-left:50px;
```

CSS Margin (cont.)

3. Margin - Shorthand property

- To shorten the code, it is possible to specify all the margin properties in one property. This is called a shorthand property.
- The shorthand property for all the margin properties is "margin":
- Example

```
margin:100px 50px;
```

CSS Margin (cont.)

3. Margin - Shorthand property (cont.)

- The margin property can have from **one to four** values.

  a. **four values**

  - `margin:25px 50px 75px 100px;`
    - top margin is 25px
    - right margin is 50px
    - bottom margin is 75px
    - left margin is 100px

CSS Margin (cont.)

3. Margin - Shorthand property (cont.)
   b. three values
      - `margin:25px 50px 75px;`
        - top margin is 25px
        - right and left margin are 50px
        - bottom margin is 75px

CSS Margin (cont.)

3. Margin - Shorthand property (cont.)

   c. two values

   - `margin:25px 50px;`
     - top and bottom margins are 25px
     - right and left margins are 50px

   d. one value

   - `margin:25px;`
     - all four margins are 25px

CSS Padding

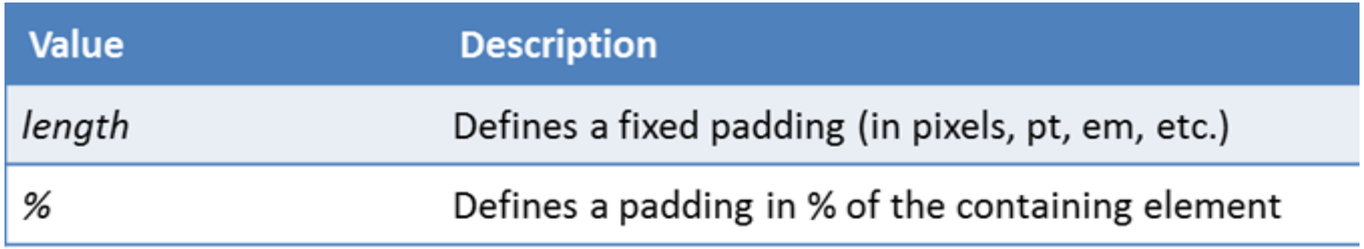
The CSS padding properties define the space between the element border and the element content.

1. Padding

   - The padding clears an area around the content (inside the border) of an element. The padding is affected by the background color of the element.
   - The top, right, bottom, and left padding can be changed independently using separate properties. A shorthand padding property can also be used, to change all paddings at once.

CSS Padding (cont.)

1. Padding (cont.)
    - Possible Values



Value	Description
length	Defines a fixed padding (in pixels, pt, em, etc.)
%	Defines a padding in % of the containing element

CSS Padding (cont.)

2. Padding - Shorthand property

- To shorten the code, it is possible to specify all the padding properties in one property. This is called a shorthand property.
- The shorthand property for all the padding properties is "padding":
- Example

```
padding:25px 50px;
```

CSS Padding (cont.)

2. Padding - Shorthand property (cont.)

- The padding property can have from **one to four** values.

  a. **four values**

     - `padding:25px 50px 75px 100px;`
       - top padding is 25px
       - right padding is 50px
       - bottom padding is 75px
       - left padding is 100px

CSS Padding (cont.)

2. Padding - Shorthand property (cont.)

   b. three values

   - `padding:25px 50px 75px;`
     - top padding is 25px
     - right and left paddings are 50px
     - bottom padding is 75px

CSS Padding (cont.)

2. Padding - Shorthand property (cont.)

   c. two values

      - `padding:25px 50px;`
        - top and bottom paddings are 25px
        - right and left paddings are 50px

   d. one value

      - `padding:25px;`
        - all four paddings are 25px

CSS Display

The display property specifies if/how an element is displayed, and the visibility property specifies if an element should be visible or hidden.

1. Hiding an Element

- Hiding an element can be done by setting
  - the **display** property to "**none**"
  - or the **visibility** property to "**hidden**".
- However, notice that these **two** methods produce different results.

CSS Display (cont.)

1. Hiding an Element (cont.)
   - visibility:hidden
     - hides an element,
     - but it will still take up the same space as before.
     - The element will be hidden, but still affect the layout.
     - Example

```
        h1.hidden {visibility:hidden;}
```

CSS Display (cont.)

1. **Hiding an Element (cont.)**
   - display:none
     - hides an element,
     - and it will not take up any space.
     - The element will be hidden, and the page will be displayed as if the element is not there.
     - Example

```
h1.hidden {display:none;}
```

CSS Display (cont.)

2. Block and Inline Elements
    - Block Elements
        - A block element is an element that takes up the full width available, and has a line break before and after it.
        - Example

```
<h1>, <p>, <div>
```

CSS Display (cont.)

2. Block and Inline Elements (cont.)
   - Inline Elements
     - An inline element only takes up as much width as necessary, and does not force line breaks.
     - Example

```
<span>, <a>
```

CSS Display (cont.)

3. Changing How an Element is Displayed

- Changing an **inline element** to a **block element**, or vice versa, can be useful for making the page look a specific way, and still follow web standards.
- The following example displays list items as inline elements:
  - Example `p {display:inline;}`
- The following example displays span elements as block elements:
  - Example `span {display:block;}`

CSS Display (cont.)

4. The display: inline-block value

- Compared to **display: inline**, the major difference is that **display: inline-block** allows to set a width and height on the element.
- Also, with **display: inline-block**, the top and bottom margins/paddings are respected, but with **display: inline** they are not.
- Compared to **display: block**, the major difference is that **display: inline-block** does not add a line-break after the element, so the element can sit next to other elements.

CSS Display (cont.)

5. The display: inline-block usage
    - Use inline-block to create navigation links
        - One common use for display: **inline-block** is to create horizontal navigation links.
    - Use with Grid of Boxes
        - It has been possible for a long time to create a grid of boxes that fills the browser width and wraps nicely (when the browser is resized), by using the **float** property.

CSS Positioning

1. Positioning
    - The CSS positioning properties allow you to position an element. It can also place an element behind another, and specify what should happen when an element's content is too big.
    - Elements can be positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first. They also work differently depending on the positioning method.

CSS Positioning (cont.)

1. Positioning (cont.)
   - There are five different positioning methods such as :
     a. static,
     b. fixed,
     c. relative,
     d. absolute
     e. and sticky.

CSS Positioning (cont.)

1. Positioning (cont.)
   a. Static Positioning
      - HTML elements are positioned static by default. A static positioned element is always positioned according to the normal flow of the page.
      - Static positioned elements are not affected by the top, bottom, left, and right properties.

CSS Positioning (cont.)

1. Positioning (cont.)
   b. Fixed Positioning
      - An element with fixed position is positioned relative to the browser window.
      - It will not move even if the window is scrolled:
      - Example

```
p.pos_fixed {
position:fixed;
top:30px;
right:5px;
}
```

CSS Positioning (cont.)

1. Positioning (cont.)
   b. Fixed Positioning (cont.)
      - Fixed positioned elements are removed from the normal flow. The document and other elements behave like the fixed positioned element does not exist.
      - Fixed positioned elements can overlap other elements.

CSS Positioning (cont.)

1. Positioning (cont.)
   c. Relative Positioning
      - A relative positioned element is positioned relative to its normal position.
      - Example

```
h2.pos_left {
     position:relative;
     left:-20px;
}
```

```
h2.pos_right {
     position:relative;
     left:20px;
}
```

CSS Positioning (cont.)

1. Positioning (cont.)
   c. Relative Positioning (cont.)
      - The content of relatively positioned elements can be moved and overlap other elements, but the reserved space for the element is still preserved in the normal flow.
      - Example

```
h2.pos_top{     position:relative;
                top:-50px; }
```

      - Relatively positioned elements are often used as container blocks for absolutely positioned elements.

CSS Positioning (cont.)

1. Positioning (cont.)
    d. Absolute Positioning
    - An absolute position element is positioned relative to the first parent element that has a position other than static. If no such element is found, the containing block is <html>:
    - Example

```
    h2 {
        position:absolute;
        left:100px;
        top:150px;
}
```

CSS Positioning (cont.)

1. Positioning (cont.)
   d. Absolute Positioning (cont.)
      - Absolutely positioned elements are removed from the normal flow. The document and other elements behave like the absolutely positioned element does not exist.
      - Absolutely positioned elements can overlap other elements.

CSS Positioning (cont.)

1. Positioning (cont.)
   e. Sticky Positioning
      - A position: sticky is a CSS property that allows an element to be positioned based on the user's scroll position.
      - The element is treated as relatively positioned until it crosses a specified point during scrolling, at which point it is treated as fixed. This behavior combines aspects of both position: relative and position: fixed.

CSS Positioning (cont.)

1. Positioning (cont.)
    e. Sticky Positioning (cont.)
        - Example

```
.box-1 {
        position: sticky;
        top: 0px;
    }
```

CSS Positioning (cont.)

2. Overlapping Elements (z-index)

    - When elements are positioned outside the normal flow, they can overlap other elements.
    - The z-index property specifies the stack order of an element (which element should be placed in front of, or behind, the others).
    - An element can have a positive or negative stack order.

CSS Positioning (cont.)

2. Overlapping Elements (z-index)

- Example

```
img {   position:absolute;
        left:0px;
        top:0px;
        z-index:-1; }
```

- An element with greater stack order is always in front of an element with a lower stack order.

***Note**: If two positioned elements overlap without a z-index specified, the element positioned last in the HTML code will be shown on top.*

CSS Floating

1. What is CSS Float?

- With CSS float, an element can be pushed to the left or right, allowing other elements to wrap around it.
- Float is very often used for images, but it is also useful when working with layouts.

CSS Floating (cont.)

2. <u>How Elements Float</u>
   - Elements are floated horizontally, this means that an element can only be floated left or right, not up or down.
   - A floated element will move as far to the left or right as it can. Usually this means all the way to the left or right of the containing element.

CSS Floating (cont.)

2. How Elements Float (cont.)
    - The elements after the floating element will flow around it. The elements before the floating element will not be affected. If an image is floated to the right, a following text flows around it, to the left:
    - Example

```
img { float:right; }
```

CSS Floating (cont.)

3. Floating Elements Next to Each Other

- If you place several floating elements after each other, they will float next to each other if there is room.
- Here we have made an image gallery using the float property:
- Example

```
.thumbnail {          float:left;
                width:110px;
                height:90px;
                margin:5px;            }
```

CSS Floating (cont.)

4. Turning off Float - Using Clear

- Elements after the floating element will flow around it. To avoid this, use the clear property.
- The clear property specifies which sides of an element other floating elements are not allowed.
- Add a text line into the image gallery, using the clear property:
- Example

```
.text_line { clear:both; }
```

CSS Align

1. Center Aligning Using the margin Property

- Block elements can be center-aligned by setting the left and right margins to "auto".
- Setting the left and right margins to auto specifies that they should split the available margin equally. The result is a centered element
- Example

```
.center { margin-left:auto;
          margin-right:auto;
          width:70%;
          background-color:#b0e0e6; }
```

CSS Align (cont.)

2. Left and Right Aligning Using the position Property

- One method of aligning elements is to use absolute positioning:
- Example:

```
.right {   position:absolute;
           right:0px;
           width:300px;
           background-color:#b0e0e6;
        }
```

***Note**: Absolute positioned elements are removed from the normal flow, and can overlap elements.*

CSS Align (cont.)

3. Left and Right Aligning Using the float Property

- One method of aligning elements is to use the float property:
- Example:

```
.right {
        float:right;
        width:300px;
        background-color:#b0e0e6;
}
```

CSS Pseudo-classes

- **A pseudo-class** is used to define a special state of an element.
- **Syntax:**

```
selector:pseudo-class {
    css declaration;
}
```

or

```
selector.class:pseudo-class {
    css declaration;
}
```

CSS Pseudo-classes (cont.)

1. Anchor Pseudo-classes
   - Links can be displayed in different ways in a CSS-supporting browser:
   - Example:

```
a:link {color:#FF0000;}       /* unvisited link */
a:visited {color:#00FF00;}  /* visited link */
a:hover {color:#FF00FF;}  /* mouse over link */
a:active {color:#0000FF;}  /* selected link */
```

CSS Pseudo-classes (cont.)

2. Pseudo-classes and CSS Classes

- Pseudo-classes can be combined with CSS classes:
- Example:

```
a.red:visited {color:#FF0000;}
<a class="red" href="css_syntax.asp">CSS Syntax</a>
```

- If the link in the example above has been visited, it will be displayed in red.

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:active	a:active	selects the active link
:checked	input:checked	selects every checked <input> element
:disabled	input:disabled	selects every disabled <input> element
:empty	p:empty	selects every <p> element that has no children
:enabled	input:enabled	selects every enabled <input> element
:valid	input:valid	selects all <input> elements with a valid value
:visited	a:visited	selects all visited links

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:first-child	p:first-child	selects every <p> elements that is the first child of its parent
:first-of-type	p:first-of-type	selects every <p> element that is the first <p> element of its parent
:focus	input:focus	selects the <input> element that has focus
:hover	a:hover	selects links on mouse over
:in-range	input:in-range	selects <input> elements with a value within a specified range

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:invalid	input:invalid	selects all <input> elements with an invalid value
:lang(*language*)	p:lang(it)	selects every <p> element with a lang attribute value starting with "it"
:last-child	p:last-child	selects every <p> elements that is the last child of its parent
:last-of-type	p:last-of-type	selects every <p> element that is the last <p> element of its parent

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:link	a:link	selects all unvisited links
:not(selector)	:not(p)	selects every element that is not a <p> element
:nth-child(n)	p:nth-child(2)	selects every <p> element that is the second child of its parent
:nth-last-child(n)	p:nth-last-child(2)	selects every <p> element that is the second child of its parent, counting from the last child

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:nth-last-of-type(n)	p:nth-last-of-type(2)	selects every <p> element that is the second <p> element of its parent, counting from the last child
:nth-of-type(n)	p:nth-of-type(2)	selects every <p> element that is the second <p> element of its parent
:only-of-type	p:only-of-type	selects every <p> element that is the only <p> element of its parent
:only-child	p:only-child	selects every <p> element that is the only child of its parent

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:optional	input:optional	selects <input> elements with no "required" attribute
:out-of-range	input:out-of-range	selects <input> elements with a value outside a specified range
:read-only	input:read-only	selects <input> elements with a "readonly" attribute specified
:read-write	input:read-write	selects <input> elements with no "readonly" attribute

CSS Pseudo-classes (cont.)

All CSS Pseudo Classes

Selector	Example	Example Description
:required	input:required	selects <input> elements with a "required" attribute specified
:root	root	selects the document's root element
:target	#news:target	selects the current active #news element (clicked on a url containing that anchor name)

CSS Pseudo-element

- **A CSS pseudo-element** is used to style specified parts of an element.
- **Syntax:**

```
selector::pseudo-element {
    css declaration;
}
```

or

```
selector.class::pseudo-element {
    css declaration;
}
```

- **Usage:**
  - Style the first letter, or line, of an element.
  - Insert content before, or after, the content of an element.

CSS Pseudo-element (cont.)

1. The ::first-line Pseudo-element

- The ::first-line pseudo-element is used to add a special style to the first line of a text.
- It can only be applied to block-level elements.
- Example of format the first line of the text in p elements:

```
p::first-line {
    color:#ff0000;
    font-variant:small-caps;
}
```

CSS Pseudo-element (cont.)

2. The ::first-letter Pseudo-element

- The ::first-letter pseudo-element is used to add a special style to the first letter of a text.
- It can only be applied to block-level elements.
- Example of format the first letter of the text in p elements:

```
p::first-letter {
    color:#ff0000;
    font-size:xx-large;
}
```

CSS Pseudo-element (cont.)

3. The ::before Pseudo-element

- The ::before pseudo-element can be used to insert some content before the content of an element.
- The following example inserts an image before each <h1> element:
- Example

```
h1::before { content:”👍” }
```

CSS Pseudo-element (cont.)

4. The ::after Pseudo-element

- The ::after pseudo-element can be used to insert some content after the content of an element.
- The following example inserts an image after each <h1> element:
- Example

```
h1::after { content:"👍" }
```

CSS Pseudo-element (cont.)

5. Pseudo-elements and CSS Classes

- Pseudo-elements can be combined with CSS classes:
- Example

```
p.article::first-letter {color:#ff0000;}
<p class="article">A paragraph in an article</p>
```

- The example above will display the first letter of all paragraphs with class="article", in red.

CSS Media Types

By using the **@media** rule, a website can have a different layout for screen, print, mobile phone, tablet, etc.

1. Media Types

   - Some CSS properties are only designed for a certain media. For example the "voice-family" property is designed for aural user agents. Some other properties can be used for different media types.

CSS Media Types (cont.)

1. Media Types (cont.)
   - Example the "**font-size**" property
     - can be used for both screen and print media, but perhaps with different values.
     - A document usually needs a larger **font-size** on a screen than on paper, and sans-serif fonts are easier to read on the screen, while serif fonts are easier to read on paper.

CSS Media Types (cont.)

2. The @media Rule

- The **@media** rule allows different style rules for different media in the same style sheet.

```
body {   background-color: aqua;    }
@media screen and (min-width: 200px) and (max-width: 400px){
  body {  background-color: lightgreen;
   }
}
@media screen and (min-width: 400px) and (max-width: 1080px){
  body {  background-color: blueviolet;
   }
}
```

References

- Understand '+', '>' and '~' symbols in CSS Selector
    - > all tags in first level inside container
    - + only one except the first tag inside container
    - ~ all tags except the first tag inside
- CSS Selectors

Thank You