



# CSS

Dr. L.M. Jenila Livingston  
VIT Chennai

# What is CSS?



**CSS** stands for **C**ascading **S**tyle **S**heets  
Styles - define **how to display** HTML elements  
Styles are normally stored in **Style Sheets**

Cascading Style Sheets (CSS) – is a rule based language that applies **styling to your HTML elements**.

Write CSS rules for elements, and modify properties of those elements such as color, background color, width, border thickness, font size, etc.

CSS files are stored with an extension **.css**

.

# Why CSS?

- Development of large web sites, where fonts and color information were added to every single page, became a long and expensive process.
- All formatting could be removed from the HTML document, and stored in a separate CSS file.
- Since the styling is written separately in external css, the html page gets loaded faster.

# <style> type attribute

```
<style type="text/css">  
  h1 {color:red;}  
  p {color:lightblue;}  
</style>
```

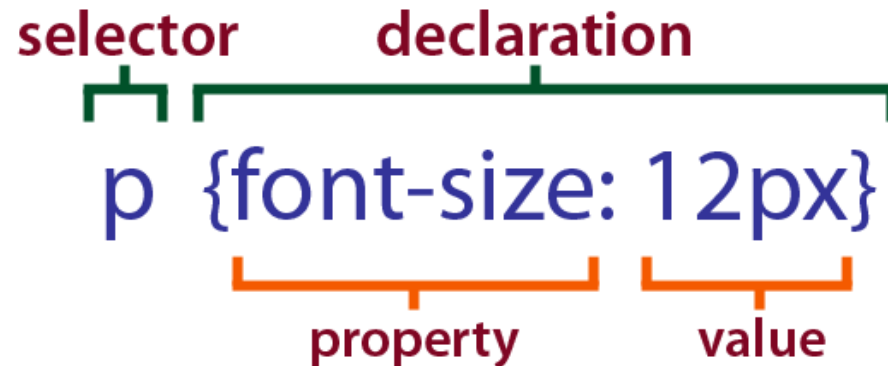


# Syntax of CSS

The CSS syntax is made up of 2 parts:

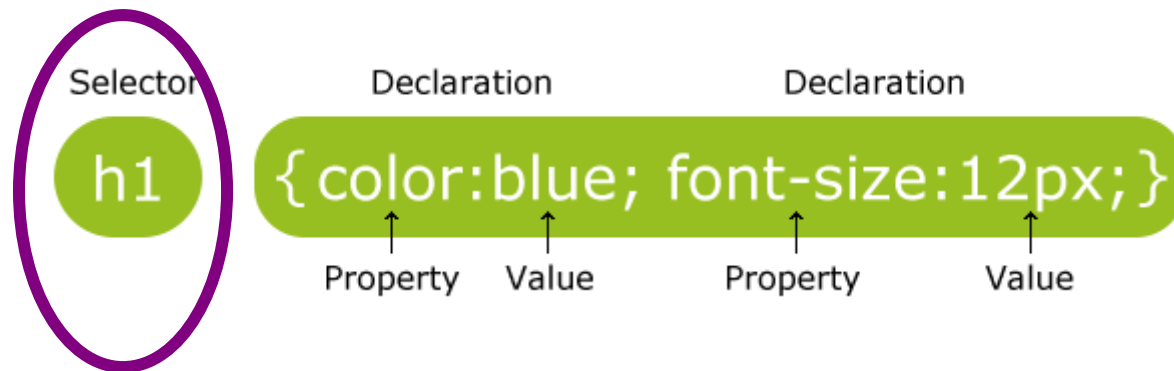
- **Selector**
- **declaration / declaration block**

- ☐ Curly Braces
- ☐ Property/attribute
- ☐ Colon
- ☐ Value
- ☐ Semicolon



# Selector

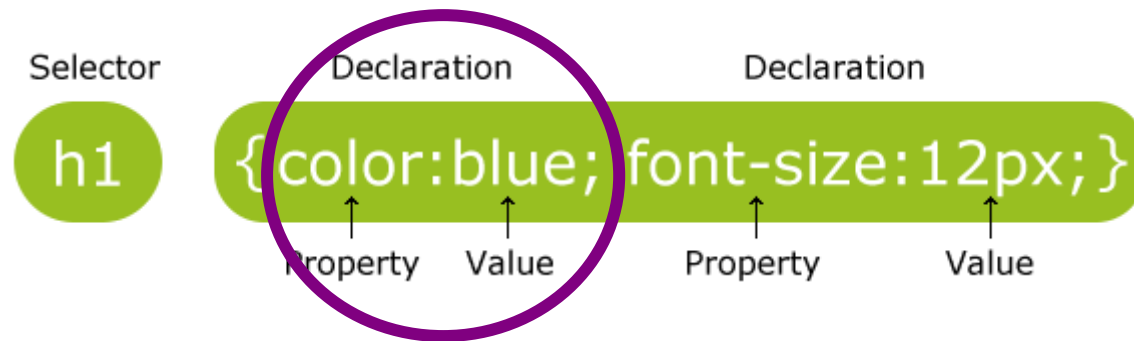
Identifies the HTML elements that the rule will be applied to,  
identified by the actual element name, e.g. <body>,  
or by other means such as **class/id** attribute values.



\*The selector is normally the HTML element you want to style

# Declaration

Each declaration includes property and value



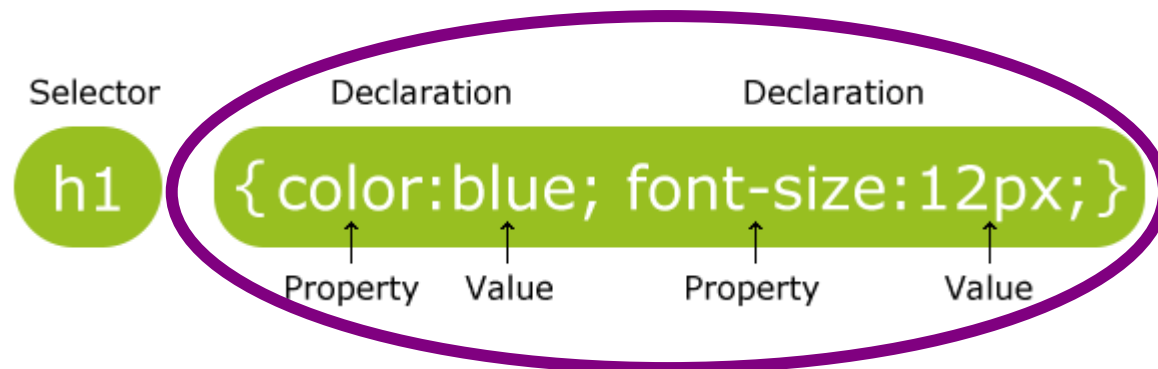
- \* Properties are separated from their respective values by **colons :**
- \* Declaration ends with **semicolon ;**

# Declaration Block

The curly braces plus their content is called a **declaration block**.

The curly braces contain the **properties** of the element you want to manipulate, and the **values** that you want to change them to.

**Multiple declaration** can be done inside the curly braces





# First CSS page

- ▶ Open Notepad
- ▶ Type the following Code

```
<html>
  <head>
    <style type="text/css">
      p {color:red; text-align:center;}
    </style>
  </head>

  <body>
    <p>Hello World!</p>
    <p>This paragraph is styled with CSS.</p>
  </body>
</html>
```

# Three Methods of CSS

- ▶ CSS is applied to a web page using three different methods:
  - Inline style
  - Internal/ Embedded style sheet
  - External style sheet

# Inline CSS

- ▶ Applies styles directly to the elements by adding declarations into the style
- ▶ For Example:

```
<p style="color: red;">
```

This is a simple paragraph and the inline style makes it red.

```
</p>
```

# Inline CSS – DIV and SPAN tag



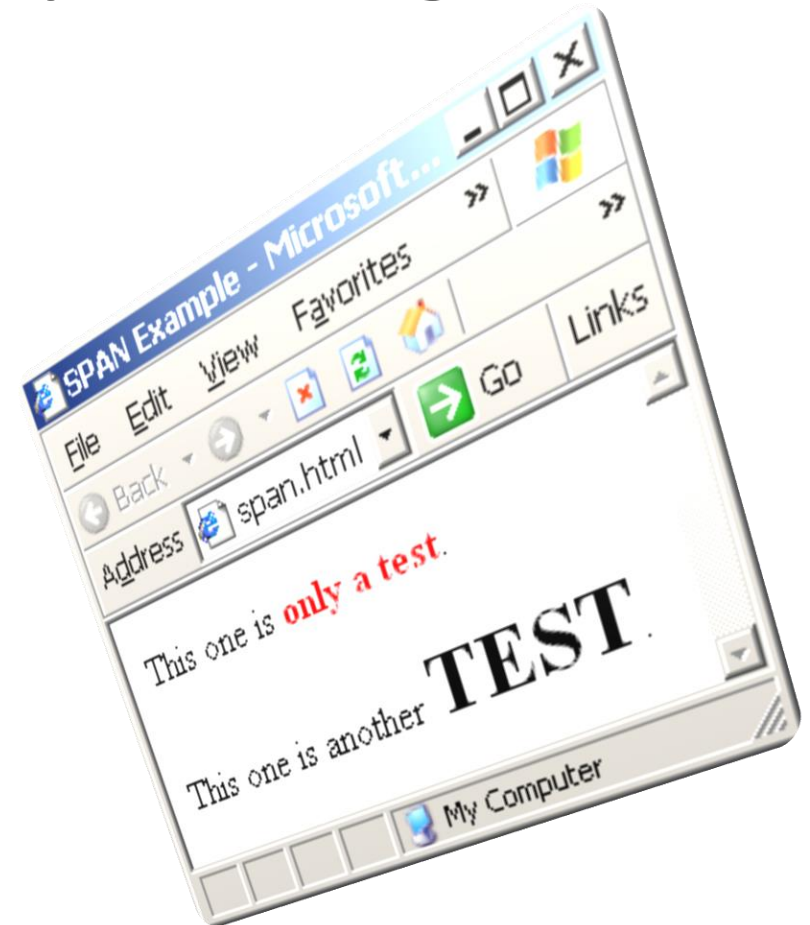
## div-and-span.html

```
<div style="font-size:24px; color:red">DIV  
example</div>
```

```
<p>This one is <span style="color:red; font-  
weight:bold">only a test</span></p>
```

# Inline CSS: The <span> Tag

- Useful for modifying a specific portion of text
  - Don't create a separate area (paragraph) in the document



span.html

```
<p>This one is <span style="color:red; font-weight:bold">only a test</span>.</p>
```

```
<p>This one is another <span style="font-size:32px; font-weight:bold">TEST</span>.</p>
```

# Internal Style Sheet

- ▶ Applies styles to HTML by placing the CSS rules inside the tag <style> inside the document tag <head>.
- ▶ For Example:

```
<head>
```

```
  <title>my page</title>
```

```
  <style type="text/css">
```

```
    p{color:red}
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
  <p>this is a simple paragraph
```

```
  </p>
```

```
</body>
```

# External Style Sheet

- Ideal when the style is applied to many pages
- 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 inside the head section:
- ```
<head>  
  <link rel="stylesheet" type="text/css"  
    href="mystyle.css">  
</head>
```

# External Style Sheet

- An external style sheet can be written in any text editor.
- The file should not contain any html tags.
- Your style sheet should be saved with a .css extension.
- An example of a style sheet file is shown below:

```
hr {color:sienna;}  
p {margin-left:20px;}  
body {background-image:url("images/back40.gif");}
```



# Types of selectors

1. Type / Element Selector
2. ID Selector
3. Class Selector
4. Universal Selector
5. Group Selector
6. Descendant Selector
7. Child Selector
8. Sibling Selector
9. Attribute Selector
10. Pseudo classes

# 1. Type Selector

- This matches and selects all the elements with the given name.

- Syntax:

**element { style properties }**

- *CSS code:*

```
h3 { color: blue; }
```

```
// all elements with h3
```

```
h1 { color: green; }
```

```
// all elements with h1
```

*HTML code:*

```
<body>
```

```
  <h1> This is header 1 tag...</h1>
```

```
  <h3> This is header 3 tag...</h3>
```

```
</body>
```

# Full width table

```
<!DOCTYPE html>
<html> <head>
<style>
table {
  width: 100%;
  border: 3px solid;
}

th, td {
  text-align: left;
  padding: 16px;
}
</style>
</head>
<body>
<h2>Full-width Table</h2>
<p>Use width: 100% to create a full-width table:</p>
```

```
<table>
  <tr>
    <th>Name</th>
    <th>Points</th>
  </tr>
  <tr>
    <td>Eve</td>
    <td>94</td>
  </tr>
  <tr>
    <td>Adam</td>
    <td>78</td>
  </tr>
</table>
```

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

```
table {
  width: 100%;
}
```

## Full-width Table

Use width: 100% to create a full-width table:

Name	Points
Eve	94
Adam	78

## 2. Id Selector

- This selector matches the id value rather than the element name. This is written with a **# symbol** preceding the id value.
- This can be used to refer to only one element.

- Syntax:

**#element\_id {style properties}**

- *CSS code:*

**#myid {color:blue; font-size:12px;}**

*HTML code:*

*<body>*

*<h3 id="myid"> CSS H3 Id Selector </h3>*

*</body>*

## 3. Class Selector

- A Class selector searches for the class name in each tag rather than the element name.
- There can be more than one element referring the same class.
- This is written with a **dot (.)** symbol preceding the class name.
- Syntax:  
**.myclass {style properties}**

- ***CSS code:***

**.myid, h1.myid {color:blue; font-size:12px;}**

***HTML code:***

***<body>***

***<h3 class="myid"> CSS H3 Id Selector </h3>***

***<h1 class="myid"> CSS H1 Id Selector </h1>***

***</body>***

## 4. Universal Selector

- The universal selector (\*) selects all the elements in the given page without any consideration of the element tags.
- This selector is used when there is a need for a common style to be applied for the entire document.
- Syntax:

**\*{style properties}**

- **CSS code:**

```
* {color:blue; font-size:12px;}
```

**HTML Code:**

```
<body>
```

```
    <h3> CSS H3 Id Selector </h3>
```

```
    <h1> CSS H1 Id Selector </h1>
```

```
</body>
```

Here, the entire document content takes a blue color & font-size of 12 pixel

# 5. Group Selector

- This selector is used to group all elements with same styling properties.
- **Comma** is used as a delimiter between elements.
- Syntax:

**element1, element2,..,elementn {style properties}**

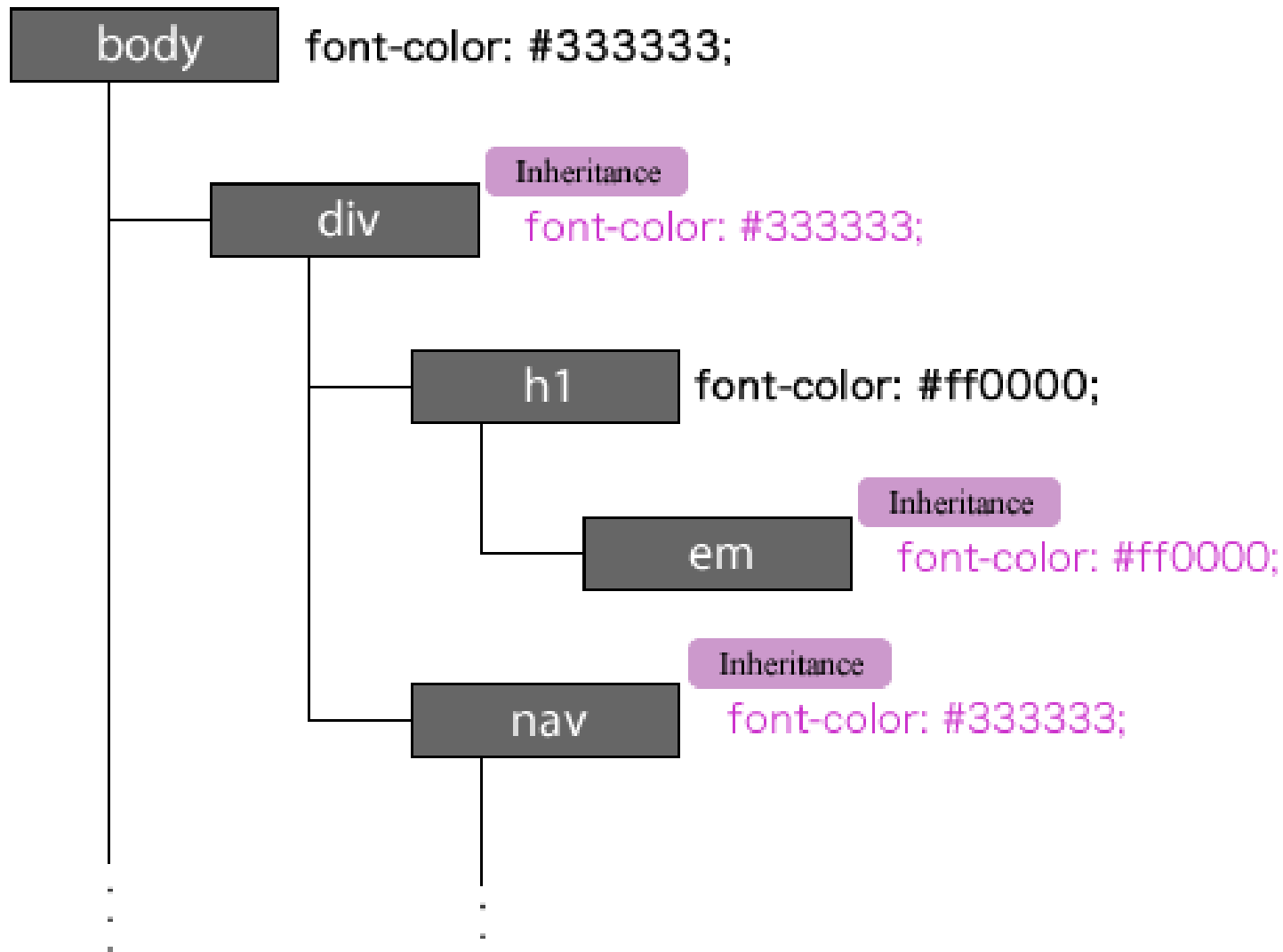
```
h1, .link, #top-link {font-weight: bold}
```

- **CSS code:**  
`h3, h1 {color:blue; font-size:12px;} // both h3 and h1 tags are grouped to have same style`

**HTML Code:**

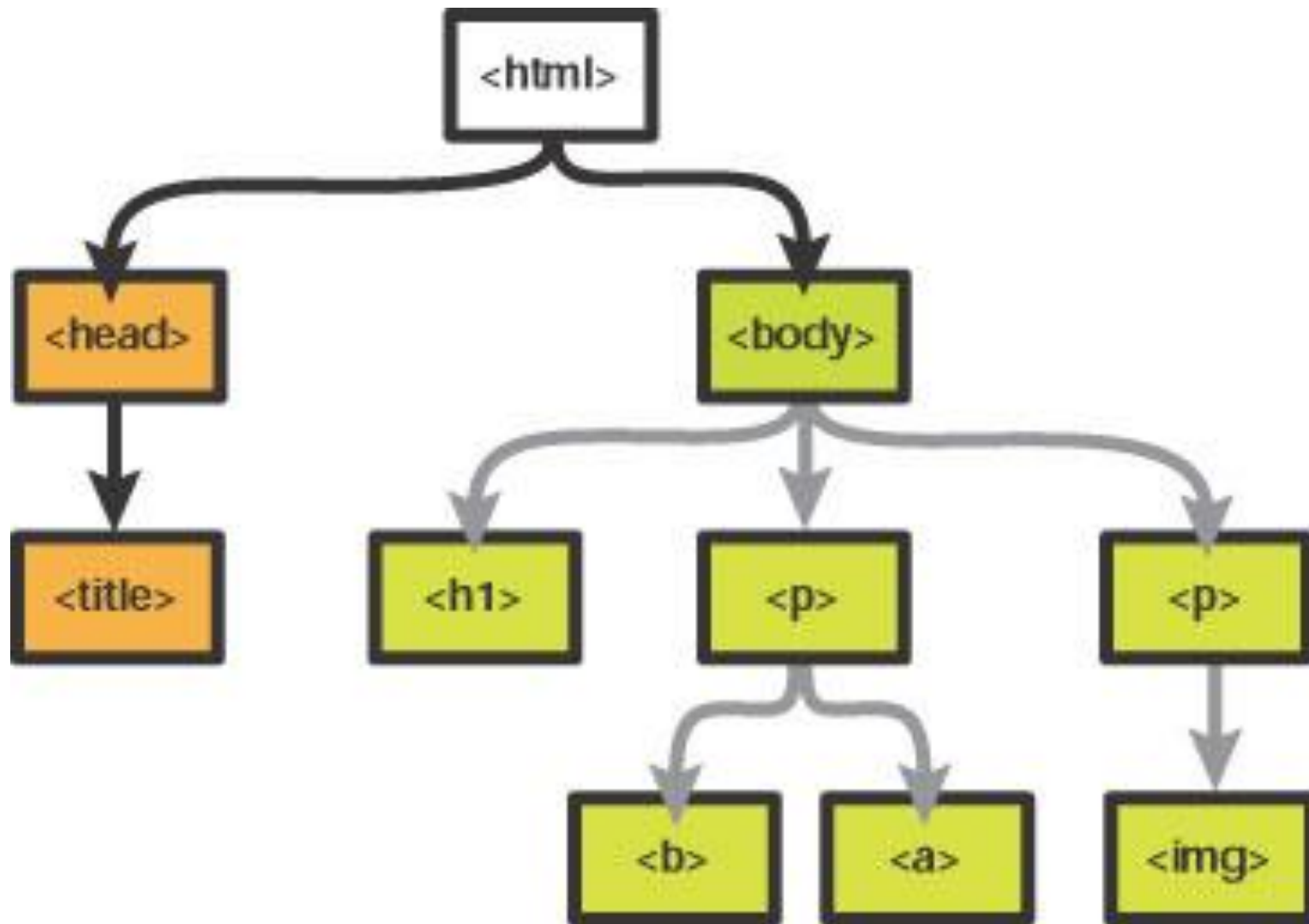
```
<body>  
  <h3> CSS H3 Id Selector </h3>  
  <h1> CSS H1 Id Selector </h1>  
</body>
```

# Inheritance





# Inheritance



# CSS Combinators

- Descendant selector (space)
- Child selector (>)
- Sibling selector (+)

## 6. Descendant Selector

- This selector represents the selection element to be a descendant of another element.
- Both the elements are specified with a **space** as a delimiter.
- Syntax:  
**element descendant element {style properties}**

- **CSS code:**

```
p h1 {color:blue; font-size:12px;}
```

### **HTML Code:**

```
<body>
```

```
<p>
```

```
    Inside paragraph code...
```

```
    <h1> This is header 1 inside p tag... </h1> // style applied
```

```
</p>
```

```
<h1> This is header 1 outside p tag....</h1> // style not applied
```

```
</body>
```

**Note:** Here, the selector style is applied for `<h1>` inside `<p>` tag & not for `<h1>` outside `<p>` tag

# 7. Child Selector

- The child selector is similar to that of a descendant selector tag except that the styling is applied to an element which is just the **direct descendant** of the given other element.
- Here, both the tags are separated by a > symbol.

Syntax:

**element > descendant element {style properties}**

- **CSS code:**

```
p > h1 {color:blue; font-size:12px;}
```

**HTML Code:**

```
<body>
```

```
<p>
```

Inside paragraph code...

```
<h1> This is header 1 inside p tag... </h1> // style applied
```

```
<section>
```

```
<h1> This is header 1 outside p tag....</h1> // style not applied
```

```
</section>
```

```
</p>
```

```
</body>
```

Note: Here, though both the <h1> tags are inside the <p> element, the styling is applied only to the <h1> element which is a direct / immediate descendant of <p>

## 8. Sibling Selector

- + selector – used to match “next sibling”:

```
li + li {color:red;}
```

```
<ul>  
  <li>One</li>  
  <li>Two</li>  
  <li>Three</li>  
</ul>
```

- One
- Two
- Three

- This will match all siblings with class name `link` that appear immediately after `<li>` tag

```
li:first-of-type + li {color:red;}
```

- One
- Two
- Three

# 9. Attribute Selector

- This selector is used to select elements with a specified attribute listed in it. It then changes the styling according to the specification. Different forms of representing the attribute selectors are;

element [ attribute ] { style properties; } // specific attribute

element [ attribute = "value" ] { style properties; } // specific property

element [ attribute ~= "value" ] { style properties; } // contains **specific word**

element [ attribute |= "value" ] { style properties; } // specific **word start**

element [ attribute ^= "value" ] { style properties; } // specific **value begin**

element [ attribute \$= "value" ] { style properties; } // specific **value end**

element [ attribute \*= "value" ] { style properties; } // contains **specific value**

~ attribute value containing a **specified word**

For | The value has to be a **whole word**, either alone, or followed by a **hyphen( - )**

For ^ The value does not have to be a whole word!

# Attribute Selector:

## Example1

```
<!DOCTYPE html>
<html>
<head>
<style>
[title] {
  border: 5px solid yellow;
}
</style>
</head>
<body>
```

### CSS [attribute] Selector

All images with the title attribute get a yellow border.



```
<h2>CSS [attribute] Selector</h2>
```

```
<p>All images with the title attribute get a yellow border.</p>
```

```

```

```

```

```

```

```
</body>
```

```
</html>
```

# Attribute Selector:

## Example2

### CSS [attribute] Selector

All images with the title attribute with the given value get a yellow border.

```
<!DOCTYPE html>
<html>
<head>
<style>
[title='flower'] {
  border: 5px solid yellow;
}
</style>
</head>
<body>
```



```
<h2>CSS [attribute] Selector</h2>
```

```
<p>All images with the title attribute with the given value get a yellow border.</p>
```

```

```

```

```

```

```

```
</body>
```

```
</html>
```



# Attribute Selector:

## Example3

```
<!DOCTYPE html>
<html>
<head>
<style>
[title~=flower] {
  border: 5px solid yellow;
}
</style>
</head>
<body>
```

### CSS [attribute~="value"] Selector

All images with the title attribute containing the word "flower" get a yellow border.



```
<h2>CSS [attribute~="value"] Selector</h2>
```

```
<p>All images with the title attribute containing the word "flower"
get a yellow border.</p>
```

```




</body>
</html>
```

Source: [https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sel\\_attribute\\_value2](https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_attribute_value2)

# Attribute Selector:

## Example4

```
<!DOCTYPE html>
<html>
<head>
<style>
[class|=top] {
  background: yellow;
}
</style>
</head>
<body>
```

```
<h2>CSS [attribute|="value"] Selector</h2>
```

```
<h1 class="top-header">Welcome</h1>
<p class="top-text">Hello world!</p>
<p class="topcontent">Are you learning CSS?</p>

</body>
</html>
```

CSS [attribute|="value"] Selector

**Welcome**

Hello world!

Are you learning CSS?

[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sel\\_attribute\\_hyphen](https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_attribute_hyphen)

# Attribute Selector:

## Example5

```
<!DOCTYPE html>
<html>
<head>
<style>
  [class*="te"] {
    background: yellow;
  }
</style>
</head>
<body>
```

### CSS [attribute\*="value"] Selector

The first div element.

The second div element.

The third div element.

This is some text in a paragraph.

```
<h2>CSS [attribute*="value"] Selector</h2>
```

```
<div class="first_test">The first div element.</div>
<div class="second">The second div element.</div>
<div class="my-test">The third div element.</div>
<p class="mytest">This is some text in a paragraph.</p>

</body>
</html>
```

[https://www.w3schools.com/css/tryit.asp?filename=trycss\\_sel\\_attribute\\_hyphen](https://www.w3schools.com/css/tryit.asp?filename=trycss_sel_attribute_hyphen)

# 10. Pseudo-classes

- A pseudo-class is used to define a special state of an element.
- For example, it can be used to:
  - Style an element when a user mouses over it
  - Style visited and unvisited links differently
  - Style an element when it gets focus

Syntax:

**selector: pseudo-class { property:value; }**

# Selectors

- Pseudo-classes define state - Pseudo-classes
  - :hover, :visited, :active :focus

```
a:hover { color: red; }
```

**Hover: mouse over link**

**Visited: visited link**

**Active: Selected link**

# Example

- `/* unvisited link */  
a:link {  
 color: #FF0000;  
}`

```
/* visited link */  
a:visited {  
    color: #00FF00;  
}
```

```
/* mouse over link */  
a:hover {  
    color: #FF00FF;  
}
```

```
/* selected link */  
a:active {  
    color: #0000FF;  
}
```

# Example

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <style>
```

```
    a.highlight:hover {color: #ff0000;}
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
  <p><a class="highlight" href="css_syntax.asp">CSS Syntax </a> </p>
```

```
  <p><a href="default.asp">CSS Tutorial</a></p>
```

```
</body>
```

```
</html>
```

# :focus

```
<!DOCTYPE html>
<html>
<head>
<style>
input:focus {
  background-color: yellow;
}
</style>
</head>
<body>

<p>Click inside the text fields to see a yellow background:</p>

<form>
  First name: <input type="text" name="firstname"><br>
  Last name: <input type="text" name="lastname">
</form>

</body>
</html>
```



# Pseudo Elements

- A CSS pseudo-element is used to style specified parts of an element.
- For example, it can be used to:
- Style the first letter, or line, of an element
- Insert content before, or after, the content of an element

Selector	Example	Example description
<u><code>::after</code></u>	<code>p::after</code>	Insert something after the content of each <code>&lt;p&gt;</code> element
<u><code>::before</code></u>	<code>p::before</code>	Insert something before the content of each <code>&lt;p&gt;</code> element
<u><code>::first-letter</code></u>	<code>p::first-letter</code>	Selects the first letter of each <code>&lt;p&gt;</code> element
<u><code>::first-line</code></u>	<code>p::first-line</code>	Selects the first line of each <code>&lt;p&gt;</code> element
<u><code>::marker</code></u>	<code>::marker</code>	Selects the markers of list items
<u><code>::selection</code></u>	<code>p::selection</code>	Selects the portion of an element that is selected by a user

# Pseudo Elements-Example

```
<!DOCTYPE html>
<html>
<head>
<style>
p::first-letter {
  color: #ff0000;
  font-size: xx-large;
}
```

**Y**OU CAN COMBINE THE ::FIRST-LETTER AND ::FIRST-LINE PSEUDO-ELEMENTS TO ADD A SPECIAL EFFECT to the first letter and the first line of a text!

```
p::first-line {
  color: #0000ff;
  font-variant: small-caps;
}
</style>
</head>
<body>
```

```
<p>You can combine the ::first-letter and ::first-line pseudo-elements to add a
special effect to the first letter and the first line of a text!</p>
```

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

# CSS

1. Color Style
2. Font
3. Background
4. Box Model
5. User Interface
6. Animation
7. Multiple Column
8. Rainbow Text

# 1. Color Style in CSS

- CSS style that includes color:

<p style="color: red;">






- Few ways that you can set colors in CSS:
  - Keywords,
  - Hex values,
  - RGB,
  - RGBA,
  - HSL

# Colors and Formatting in CSS

- ▶ CSS Colors: **Keywords**
- ▶ Using the keywords like: red, fuchsia, yellow, blue, green you can specify what color you would like the CSS rule to display.
- ▶ For example:
- ▶ `p{color:red}`
- ▶ `h2{color:yellow}`
- ▶ There are 17 of these keyword colors you can use in CSS.

# Hex values

- hexadecimal values between 00 and FF






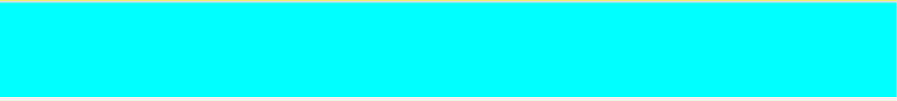
Color	HEX
	#FF0000
	#00FF00
	#0000FF
	#FFA500
	#FFFF00
	#00FFFF

<h2 style="background-color:#00FF00">

# Hex

Keyword Color	Hex
aqua	#00ffff
black	#000000
blue	#0000ff
fuchsia	#ff00ff
gray	#808080
green	#008000
lime	#00ff00
maroon	#800000
navy	#000080
olive	#808000
orange (added in CSS 2.1)	#ffa500
purple	#800080
red	#ff0000
silver	#c0c0c0
teal	#008080
white	#ffffff
yellow	#ffff00

# RGB

Color	RGB
	rgb(255,0,0)
	rgb(0,255,0)
	rgb(0,0,255)
	rgb(255,165,0)
	rgb(255,255,0)
	rgb(0,255,255)

Color	RGB
	rgb(0,0,0)
	rgb(128,128,128)
	rgb(255,255,255)

(red, green, blue) defines the intensity of the color between 0 and 255.

RGB value - rgb(255, 0, 0)

```
<h2 style="background-color:rgb(0,0,0); color:white">  
Background-color set by using  
rgb(0,0,0)  
</h2>
```



# RGB example

```
<!DOCTYPE html>
<html>
<head>
<style>
input {
  background-color: rgb(201, 76, 76);
  border-color: rgb(0,0,256);
  color: white;
}
</style>
</head>
<body>
<form>
<p>The background & border color: RGB values.</p>
<input type="text" value="Hello">
</body>
</html>
```

**Background-color:** to change the background color  
**Border-color:** To change the border color  
**Color:** to change the text color

The background & border color: RGB values.

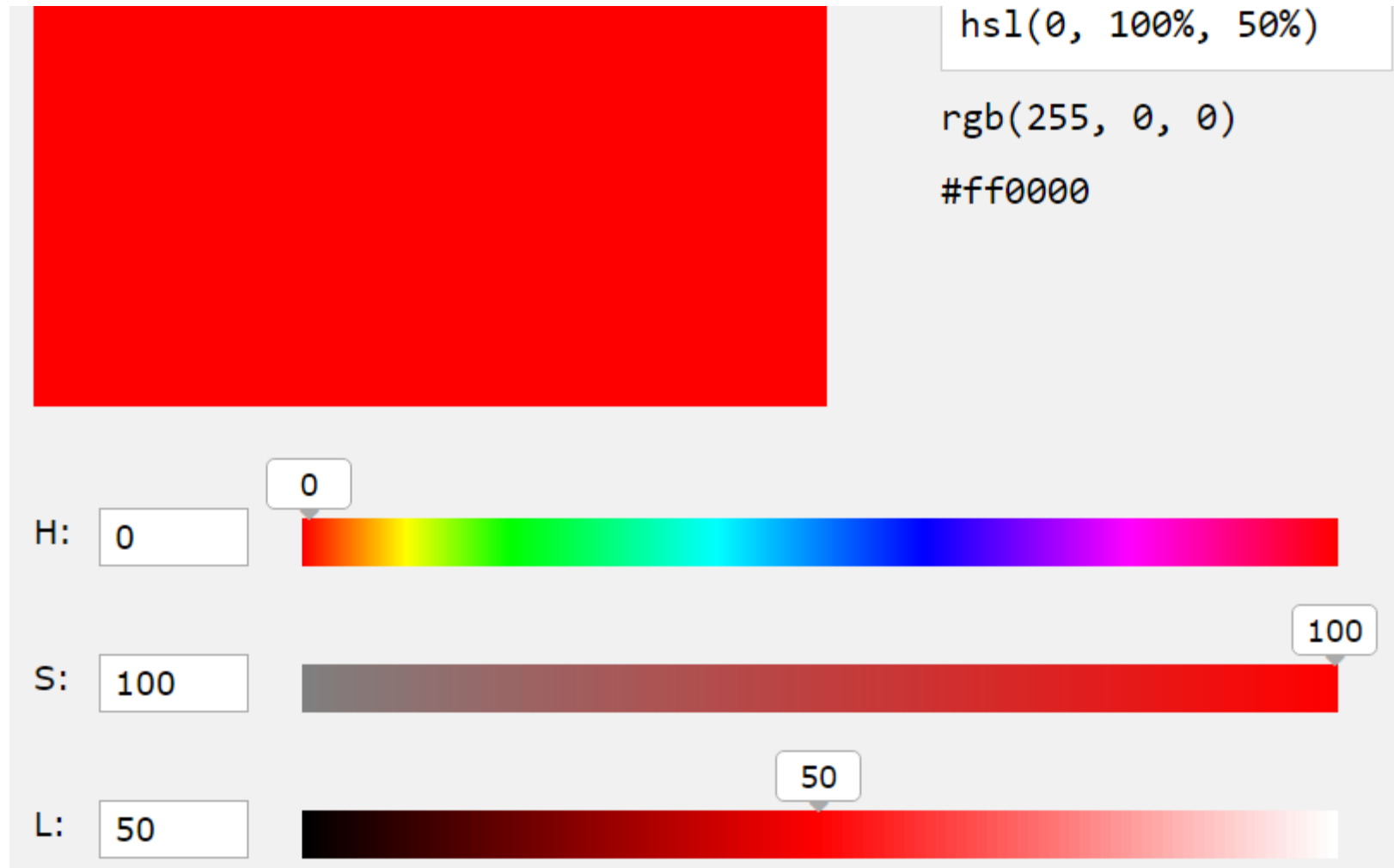
Hello

# RGBA

- An RGBA color value is specified with: `rgba(red, green, blue, alpha)`.
- The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque)

```
#p1 {background-color: rgba(255, 0, 0, 0.3);}
```

# HSL



*hsl(hue, saturation, lightness)*

# HSL

## Hue

Hue is a degree on the color wheel from 0 to 360. **0 is red, 120 is green, 240 is blue.**

## Saturation

Saturation is a percentage value; 0% means a shade of gray and 100% is the full color.

## Lightness

Lightness is also a percentage; 0% is black, 100% is white

```
<style>
```

```
div {background-color: hsl(120, 50%, 50%);}
```

```
</style>
```

## 2. Font-related CSS Properties

- **color** : specifies the color of the text
- **font-size** : xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger or numeric value
- **font-family**: comma separated font names
  - Example: verdana, sans-serif, etc.
  - The browser loads the first one that is available
  - There should always be at least one generic font
- **font-weight**: normal, bold, bolder, lighter or a number in range [100 ... 900]

# CSS Rules for Fonts

- **font-style** – styles the font
  - **Values:** normal, italic, oblique
- **text-decoration** – decorates the text
  - **Values:** none, underline, line-through, overline, blink
- **text-align** – defines the alignment of text or other content
  - **Values:** left, right, center, justify
- **text-shadow:** 2px 2px #ff0000;
- **font-variant:** normal | small-caps | initial | inherit;

# Shorthand Font Property

- **font**
  - Shorthand rule for setting multiple font properties at the same time

```
font:italic normal bold 12px/16px verdana
```

is equal to writing this:

```
font-style: italic;  
font-weight: bold;  
font-size: 12px;  
line-height: 16px;  
font-family: verdana;
```

# 3. Backgrounds

- **background-image**

- URL of image to be used as background, e.g.:

```
background-image:url("back.gif");
```

- **background-color**

- Using color and image and the same time

- **background-attachment**

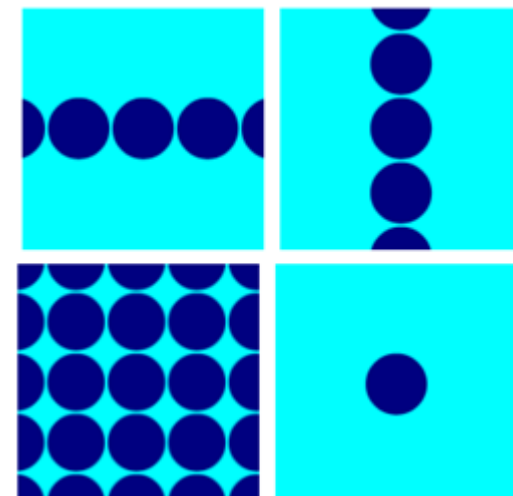
- fixed / scroll



# Backgrounds (2)

- **background-repeat**
  - repeat-x, repeat-y, repeat, no-repeat

Value	Description
background-repeat: repeat-x	The image is repeated horizontally
background-repeat: repeat-y	The image is repeated vertically
background-repeat: repeat	The image is repeated both horizontally and vertically
background-repeat: no-repeat	The image is not repeated



# Backgrounds (3)

- **background-position**: specifies **vertical and horizontal position** of the background image
  - Vertical position: top, center, bottom
  - Horizontal position: left, center, right
  - Both can be specified in percentage or other numerical values
  - Examples:

```
background-position: top left;
```

```
background-position: -5px 50%;
```

# Background Shorthand Property

- background: shorthand rule for setting background properties at the same time:

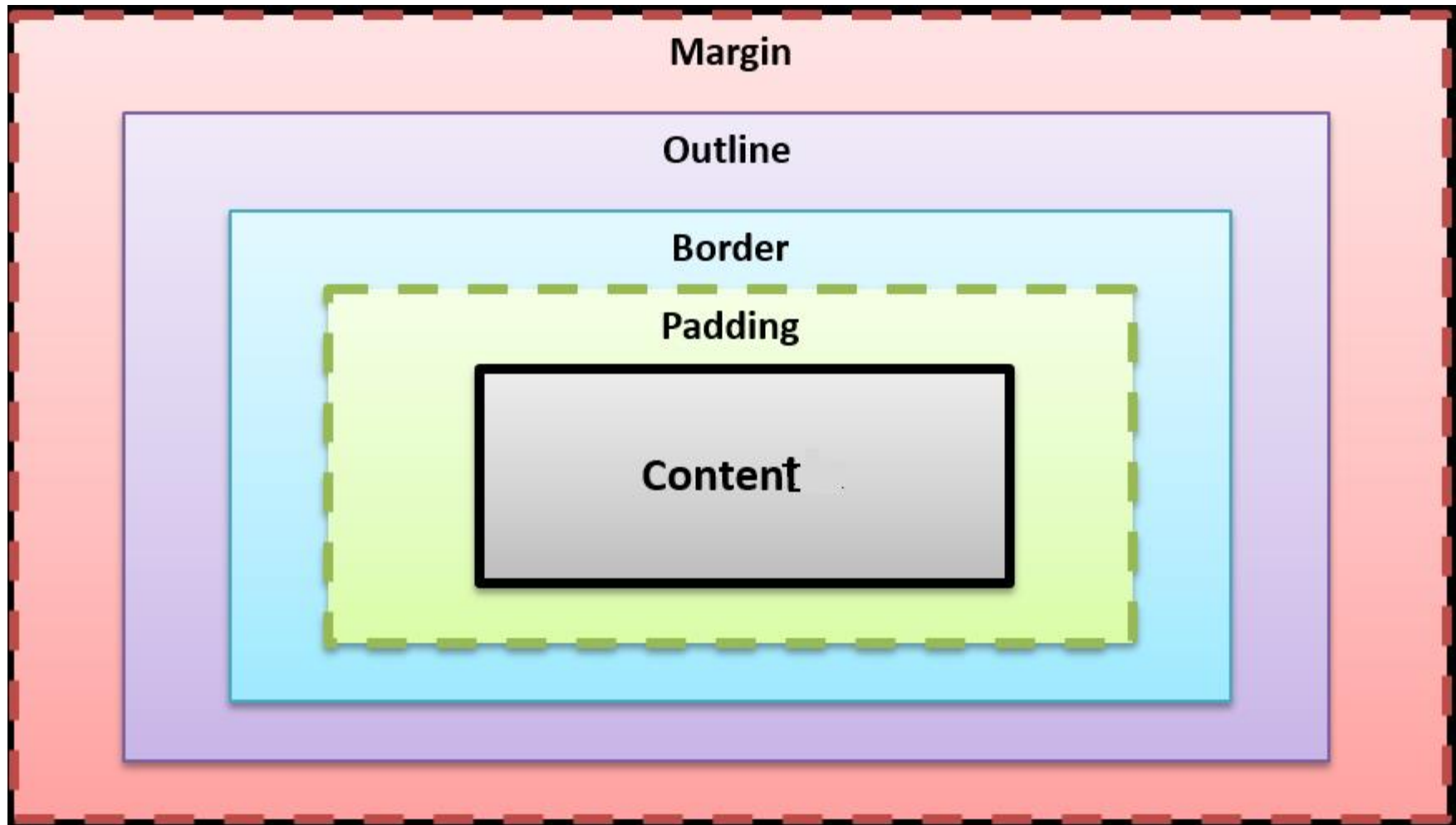
```
background: #FFF0C0 url("back.gif") no-repeat  
fixed top;
```

is equal to writing:

```
background-color: #FFF0C0;  
background-image: url("back.gif");  
background-repeat: no-repeat;  
background-attachment: fixed;  
background-position: top;
```

- Some browsers will not apply BOTH color and image for background if using shorthand rule

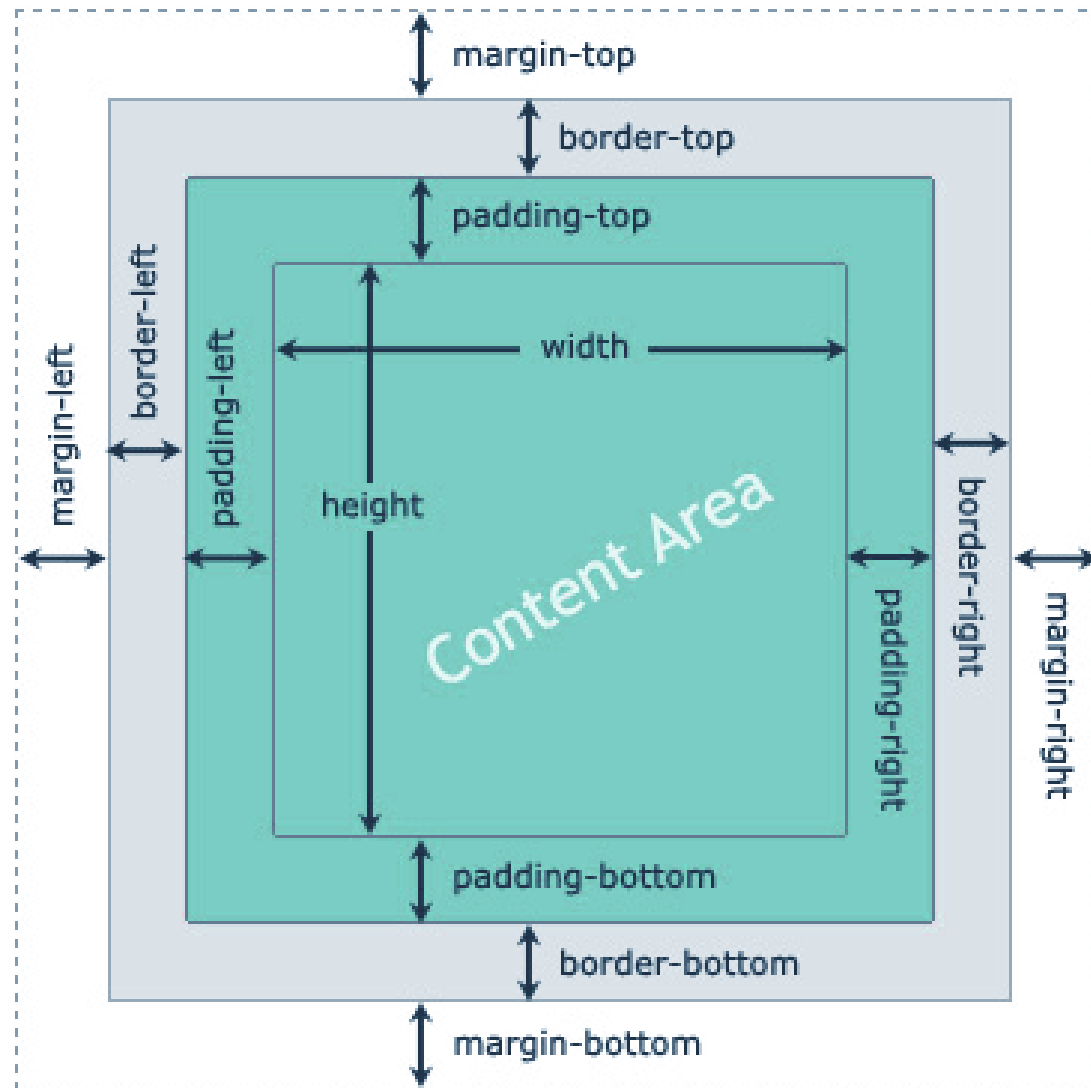
## 4. Box model (CPBOM)



# Box Elements

- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Outline** - extra border, for extra visual attention for your content
- **Margin** - Clears an area outside the border / outline. The margin is transparent

# Box Model



# Box Model Example

```
div {  
    width: 300px;  
    border: 25px solid green;  
    padding: 25px;  
    margin: 25px;  
}
```

```
<!DOCTYPE html>
<html> <head>
<style>
div {
  background-color: lightgrey;
  width: 300px;
  border: 25px solid green;
  padding: 25px;
  margin: 25px;
}
</style>
</head>
<body>
```

```
<h2>Demonstrating the Box Model</h2>
```

```
<p>The CSS box model is essentially a box that wraps around every HTML element.
It consists of: borders, padding, margins, and the actual content.</p>
```

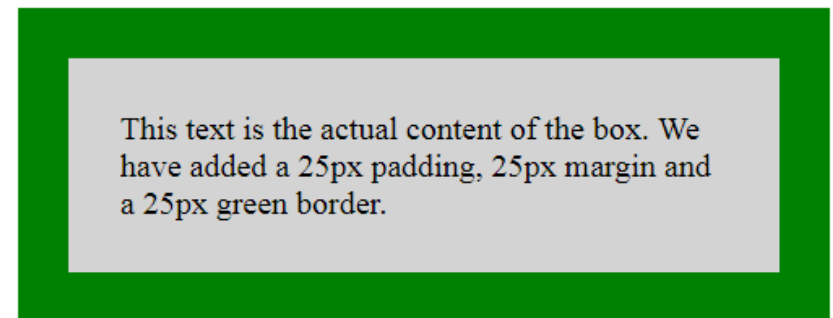
```
<div>This text is the actual content of the box. We have added a 25px padding, 25px
margin and a 25px green border. </div>
```

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

# Box model – Sample code

## Demonstrating the Box Model

The CSS box model is essentially a box that wraps around every HTML element. It consists of: borders, padding, margins, and the actual content.





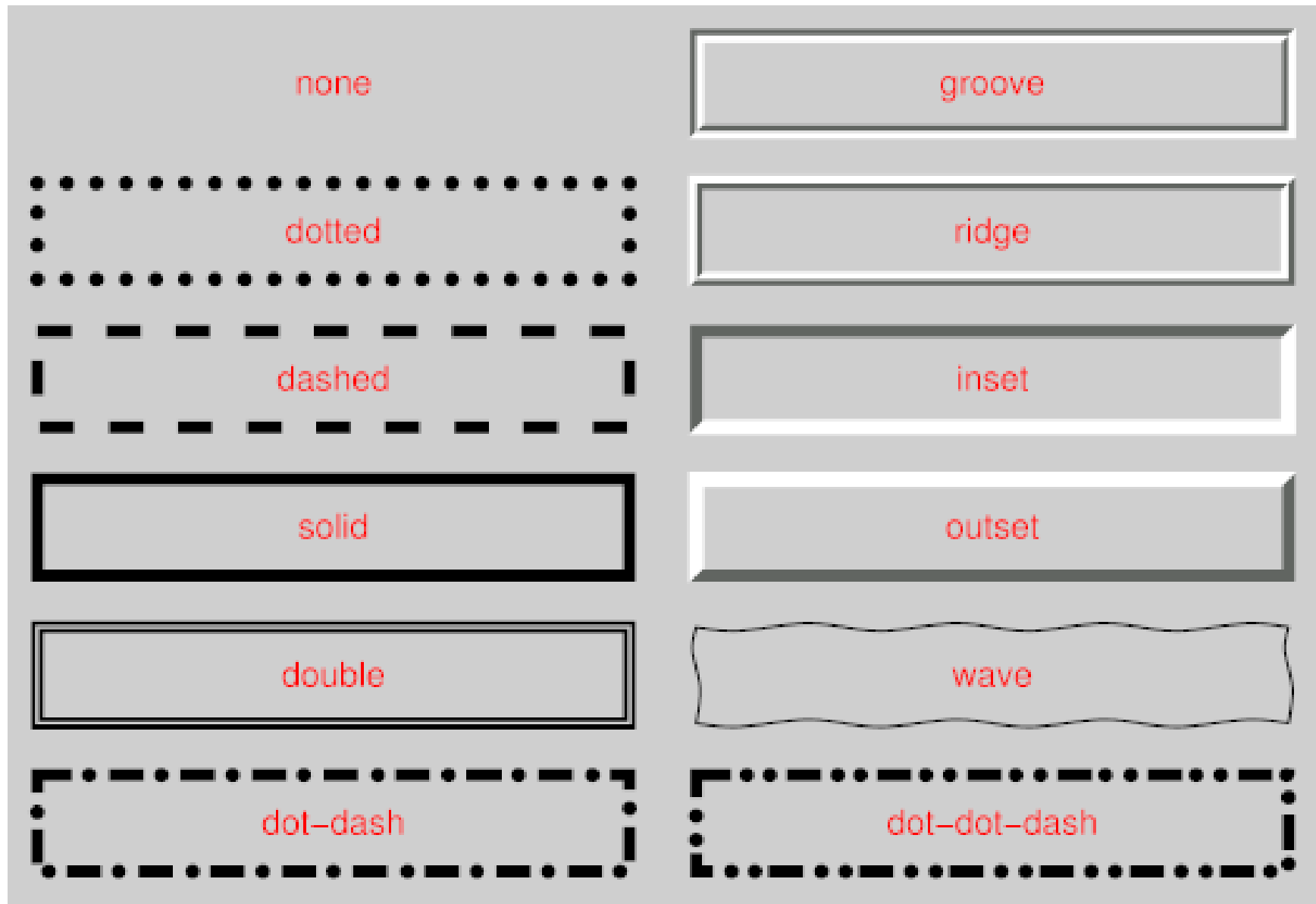
# Width and Height

- **width** – defines numerical value for the width of element, e.g. 200px
- **height** – defines numerical value for the height of element, e.g. 100px
  - By default the height of an element is defined by its content
  - Inline elements do not apply height, unless you change their **display** style.

# Borders

- **border-width:** thin, medium, thick or numerical value (e.g. 10px)
- **border-color:** color alias or RGB value
- **border-style:** none, hidden, dotted, dashed, solid, double, groove, ridge, inset, outset
- Each property can be defined separately for left, top, bottom and right
  - border-top-style, border-left-color, ...

# Border Style



# Border-radius

- `<button class="button button1">2px</button>`
- `.button1 {border-radius: 2px;}`  
`.button2 {border-radius: 4px;}`  
`.button3 {border-radius: 8px;}`  
`.button4 {border-radius: 12px;}`  
`.button5 {border-radius: 50%;}`



# Border Shorthand Property

- border: shorthand rule for setting border properties at once:

```
border: 1px solid red
```

is equal to writing:

```
border-width:1px;  
border-color:red;  
border-style:solid;
```

- Specify different borders for the sides via shorthand rules: border-top, border-left, border-right, border-bottom
- When to avoid border:0

# Padding and Margin

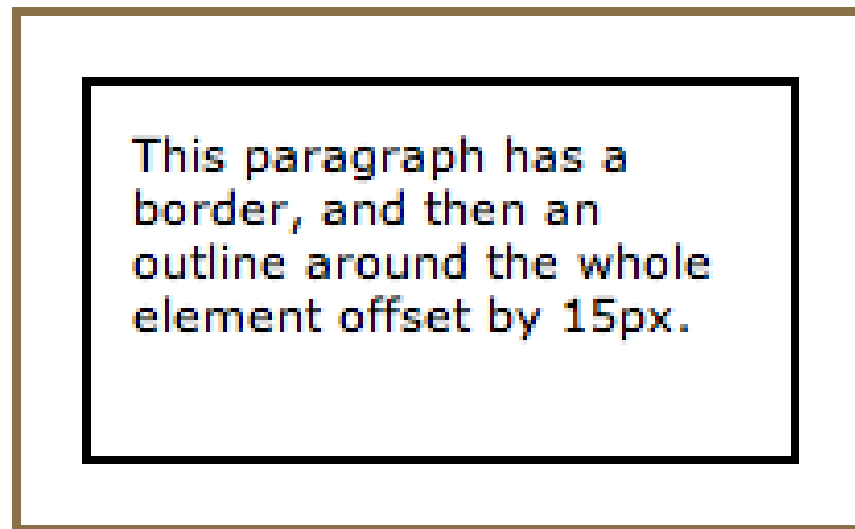
- **Padding and margin** define the spacing around the element
  - Numerical value, e.g. 10px or -5px
  - Can be defined for **each of the four sides separately** - **margin-top, padding-left, ...**
  - **margin** is the spacing outside of the border
  - **padding** is the spacing between the border and the content
  - What are collapsing margins?

# Margin and Padding: Short Rules

- `margin: 5px;`
  - Sets **all four sides** to have margin of 5 px;
- `margin: 10px 20px;`
  - **top and bottom** to 10px, **left and right** to 20px;
- `margin: 5px 3px 8px;`
  - **top** 5px, **left/right** 3px, **bottom** 8px
- `margin: 1px 3px 5px 7px;`
  - **top, right, bottom, left** (clockwise from top)
- Same for padding

# Outline offset

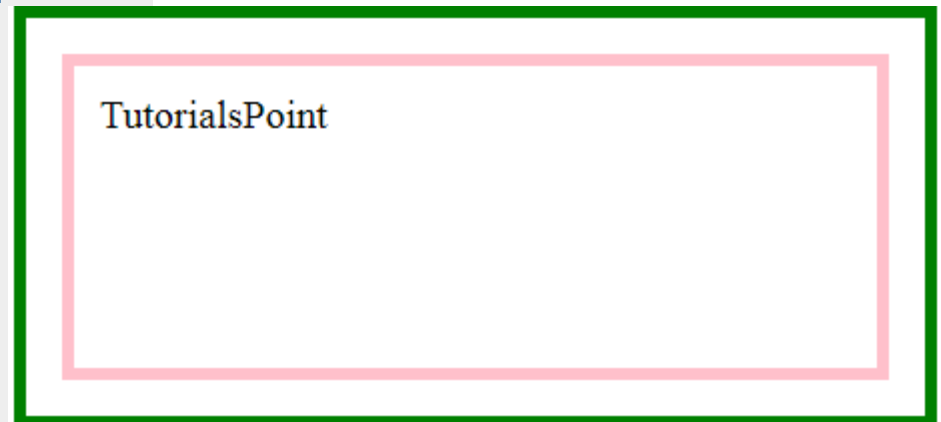
- Used to draw the behind the outline
- Out line means draw a line around the element at outside of boarder.



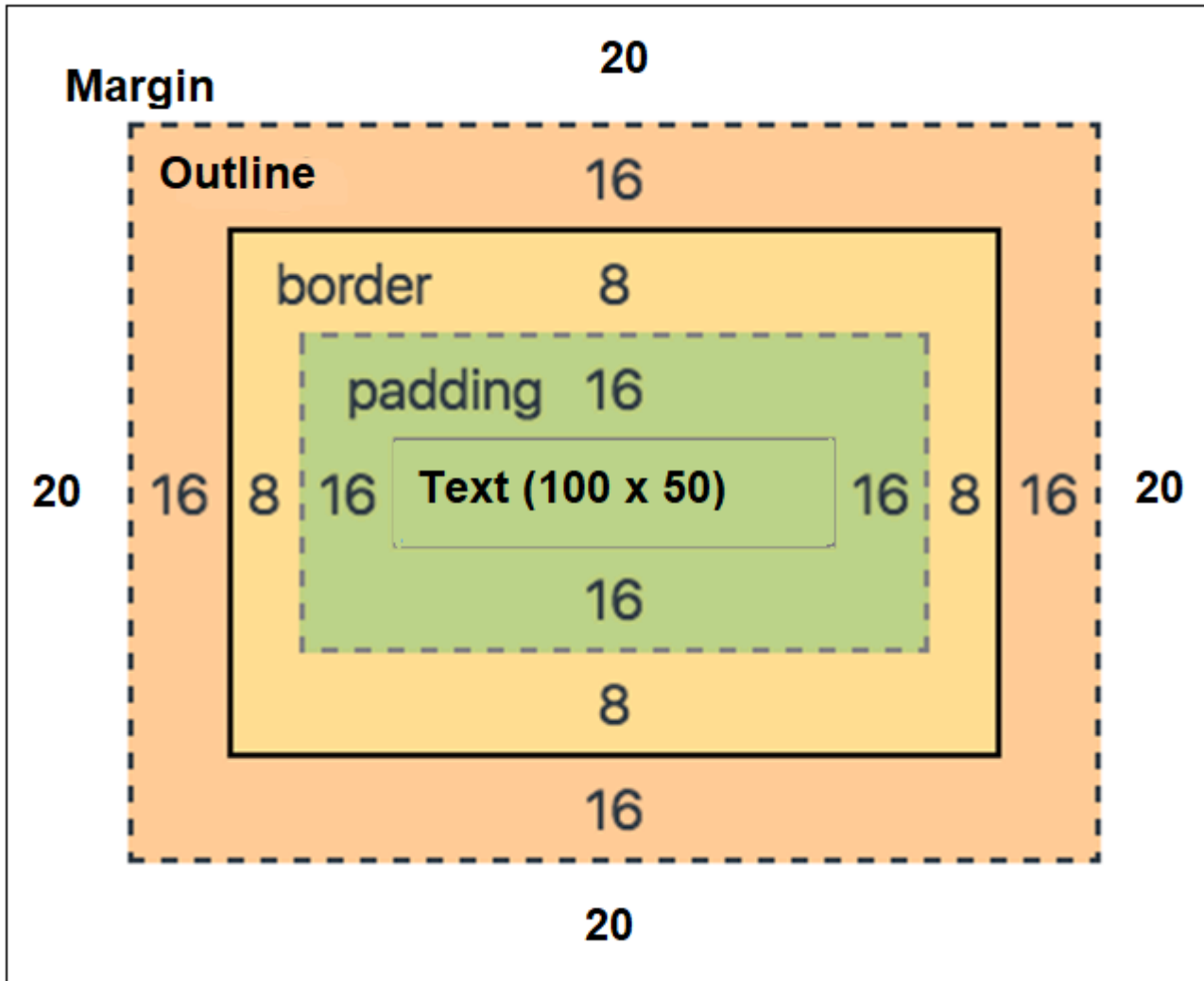


# Outline Example

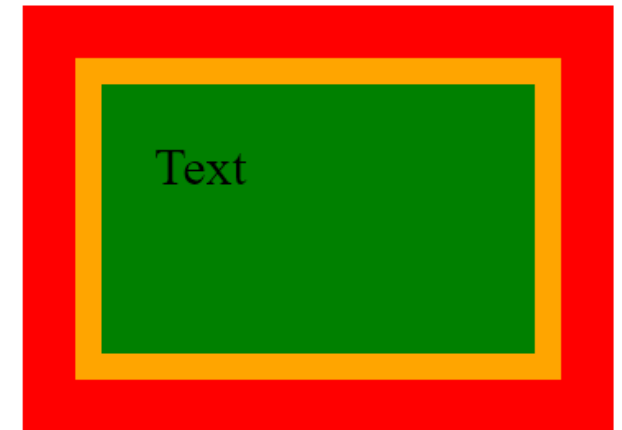
```
<html>
  <head>
    <style>
      div {
        margin: 20px;
        padding: 10px;
        width: 300px;
        height: 100px;
        border: 5px solid pink;
        outline: 5px solid green;
        outline-offset: 15px;
      }
    </style>
  </head>
  <body>
    <div>TutorialsPoint</div>
  </body>
</html>
```



# Exercise



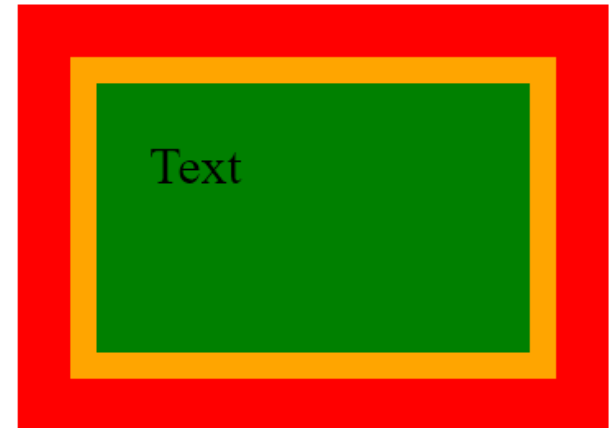
Box model



# CSS code

```
<html>
<head>
<style>
div {
  background: green;
  width: 100px;
  height: 50px;
  padding: 16px;
  margin: 20px;
  border: 8px solid orange;
  outline: 16px solid red;
}
</style>
</head>
```

Box model



```
<body>

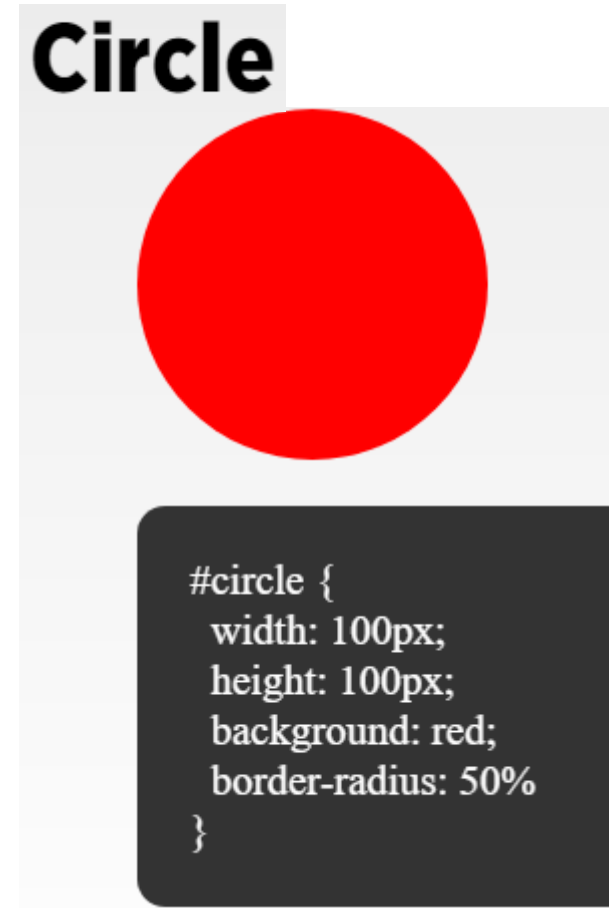
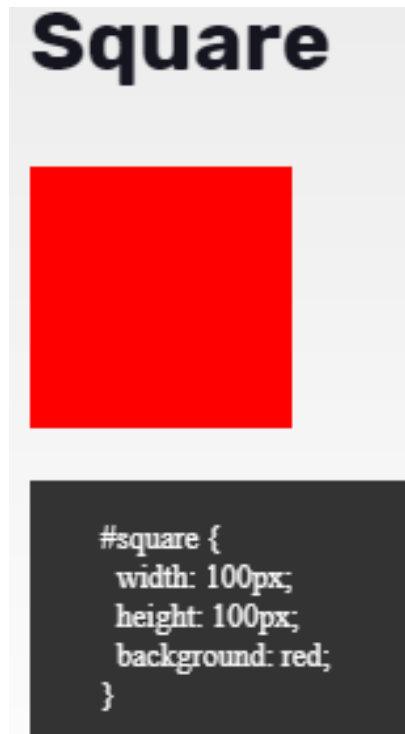
<p>Box model</p>

<div>Text</div>

</body>
</html>
```

# Shapes

- <https://css-tricks.com/the-shapes-of-css/>



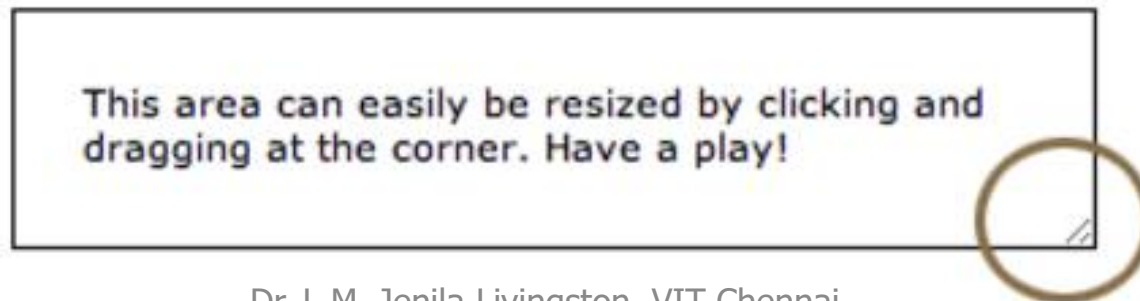
- Clipping and Masking
- <https://css-tricks.com/clipping-masking-css/>

## 5. User Interface

- The User interface of CSS includes resizing and outline-offset. The resizing provides permission of resizing the element, and in which direction .

# Resize

- Used to resize elements which are on area
- Values of resize
  - **Horizontal** : Resize the width of a element
  - **Vertical** : Resize the height of element.
  - **Both**: Resize both the height and the width of the element.
- The latest version of Safari has a feature which allows resizable text areas.



# Overflow

- The overflow property specifies what should happen if content overflows an element's box.
- This property specifies whether to clip content or to add scrollbars when an element's content is too big to fit in a specified area.

Value	Description
visible	The overflow is not clipped. It renders outside the element's box. This is default
hidden	The overflow is clipped, and the rest of the content will be invisible
scroll	The overflow is clipped, but a scroll-bar is added to see the rest of the content
auto	If overflow is clipped, a scroll-bar should be added to see the rest of the content

# Resize/ Overflow



```
<html>
  <head>
    <style>
      div {
        border: 2px solid;
        padding: 20px;
        width: 300px;
        resize: both;
        overflow: auto;
      }
    </style>
  </head>
  <body>
    <div>TutorialsPoint.com</div>
  </body>
</html>
```



# Box Sizing

- The box sizing aspect allows you to define certain elements to fit an area in a certain way
- **box-sizing:** `borderbox` | `content-box`

## Border box:

- actual width of an element
- actual height of an element

## Content box:

- $\text{width} + \text{padding} + \text{border} = \text{width of an element}$
- $\text{height} + \text{padding} + \text{border} = \text{height of an element}$



# 6. Multiple Column Layout

## A CSS3 Mutli-Column Example

This last example uses similar html as the previous examples. Here I've given the h1 to the left a column-break-after value to force anything after into a new column, which I think creates an interesting layout.

```
h1 {column-break-after: right;}
```

While I've set the value to left, using right or always accomplishes the same thing.

As with the previous example the columns are created using column-count and I've increased the column-gap to 2 'em'. If you resize your browser you'll notice the widths of the columns adapt.

However, you'll also notice the headline can't adjust its size and so at a certain point the words become hidden.

**This is an inline heading that won't span columns**

The column-rules are dotted, just for the sake of difference.

```
h1 {column-rule: 1px dotted #ccc;}
```

This layout was simple to achieve and

could work well for a magazine or news style site.

**This heading also won't span columns**

A little more care needs to go into placing the content on a static site as the column break is dynamic based on how much content there is to display.

This isn't necessarily a bad thing and it's something print designers have always had to consider and it's also a reason for thinking more flexibly about our designs.

I encourage you to play around with mutli-columns. They're easy to set up and work with and the css properties will work mostly as you'd expect.

It only took a few minutes to set up this demo page. The majority of time was spent thinking of something to say to fill up the columns with content.

Writing the code for this page was literally a few minutes at most and if you view the source code you can see how little css was necessary. Most of the code is due to the need to replicate the same properties with vendor prefixes.

# Multiple columns Layout

- Useful properties:
  - column-count : no. of columns an element is divided
  - column-fill : how to fill columns (balance | auto)
  - column-gap : space between columns (dimension)
  - column-rule-color : color of rule between columns (same as border-color)
  - column-rule-style : style of rule between columns (same as border-style)
  - column-rule-width : width of rule between columns (same as border-width)
  - column-span : span of a column
  - column-width : width of columns
  - columns : shorthand for column-width and column-count

Firefox uses prefix `–moz-` and Chrome uses `–webkit-`

# Multiple columns - example

```
div {  
    column-count : 3;  
    column-rule-color : black;  
    column-rule-style : dotted;  
    column-rule-width : 10px;  
}
```

Lorem ipsum dolor sit	● suscipit lobortis nisl ut	● dignissim qui blandit
amet, consectetur	● aliquip ex ea commodo	● praesent luptatum zzril
adipiscing elit, sed diam	● consequat. Duis autem	● delenit augue duis dolore
nonummy nibh euismod	● vel eum iriure dolor in	● te feugait nulla facilisi.
tincidunt ut laoreet	● hendrerit in vulputate	● Nam liber tempor cum
dolore magna aliquam	● velit esse molestie	● soluta nobis eleifend
erat volutpat. Ut wisi	● consequat, vel illum	● option congue nihil
enim ad minim veniam,	● dolore eu feugiat nulla	● imperdiet doming id
quis nostrud exerci	● facilisis at vero eros et	● quod mazim placerat
tation ullamcorper	● accumsan et iusto odio	● facer possim assum.

```
div {  
    columns: 40px 2; Examples:  
}
```

<http://www.quirksmode.org/css/multicolumn.html>  
<http://www.css3.info/preview/multi-column-layout/>

# 7. Animations - properties

- **@keyframes** : defines the frames of the animation
- **animation-name** : defines the animation name, used in @keyframes
- **animation-duration** : duration of the animation
- **animation-timing-function** : defines the speed of the transition; values: linear | ease | ease-in | ease-out | ease-in-out | cubic-bezier(*n,n,n,n*)
- **animation-delay** : startup delay (in seconds)
- **animation-iteration-count** : how many times the animation is played
- **animation-direction** : the direction in which animation is played (normal | reverse | alternate | alternate-reverse)
- **animation-play-state** : running or pausing the animation
- **animation** – shorthand property

Chrome uses prefix **-webkit-** and Firefox uses prefix **-moz-**

# Animation – Example 1

```
<html><head>
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
@keyframes example {
  from {background-color: red;}
  to {background-color: yellow;}
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

[https://www.w3schools.com/css/css3\\_animations.asp](https://www.w3schools.com/css/css3_animations.asp)

# Animation – Example 2

```
<style>
div {
  width: 100px;
  height: 100px;
  background-color: red;
  position: relative;
  animation-name: example;
  animation-duration: 4s;
  animation-direction: normal;
}
```

**Percentages** represent a **percentage** of the **animation** duration, 0% represents the starting point of the **animation**, 100% represents the end point, 50% represents the midpoint and so on.

```
@keyframes example {
  0%   {background-color:red; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:blue; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:red; left:0px; top:0px;}
}
```

[https://www.w3schools.com/css/tryit.asp?filename=trycss3\\_animation\\_direction](https://www.w3schools.com/css/tryit.asp?filename=trycss3_animation_direction)



# Animation – Example 3

```
<html><head>
<style>
@keyframes fontChange {
    0% {font-size:10px}
    25%,75% {font-size:300px}
    50%,100% {font-size:50px}}
div:hover {
    animation-name: fontChange;
    animation-duration: 8s; }
#come:hover {
    animation-name: fontChange;
    animation-duration: 50s;}
div {background-color:red; font-size:10px}
</style> </head>
<body><div>Hello</div>
<div id="come">Come</div></body></html>
```

## 8. Rainbow Text

```
<html><head>
```

```
<style>
```

```
.rainbow-text { background-image: linear-gradient(to left,  
violet, indigo, blue, green, yellow, orange, red); }
```

```
</style>
```

```
</head>
```


```
<body>
```

```
<h1 class="rainbow-text">Rainbow</h1>
```

```
</body>
```

```
</html>
```

The **linear-gradient()** **CSS** function creates an image consisting of a progressive transition between two or more colors along a straight **line**.  
Default: to bottom

A horizontal bar with a rainbow gradient, transitioning from red on the left to violet on the right. The text "Rainbow text" is written in a black, sans-serif font on the left side of the bar.

# Rainbow Text

```
.rainbow-text { background-image: linear-gradient(to left, violet, indigo, blue, green, yellow, orange, red);
```

```
width:150px;
```

```
-webkit-background-clip: text;
```

```
-webkit-text-fill-color: transparent; }
```

```
<h1 class="rainbow-text">Rainbow</h1>
```

The image shows the text "Rainbow text" where each letter is a different color from a rainbow spectrum. The 'R' is red, 'a' is orange, 'i' is yellow, 'n' is green, 'b' is blue, 'o' is indigo, 'w' is violet, 't' is light blue, 'e' is green, and 'x' is blue. The text is centered on a light gray background.

Chrome uses prefix `-webkit-` and Firefox uses prefix `-moz-`

Thank You