

Module 2:

Cascading Style Sheet (CSS)

Introduction

- **CSS** stands for **C**ascading **S**tyle **S**heets
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
- CSS **saves a lot of work.**

It can control the layout of multiple web pages all at once

- External stylesheets are stored in **CSS files**

Introduction (Contd.)

Why Use CSS?

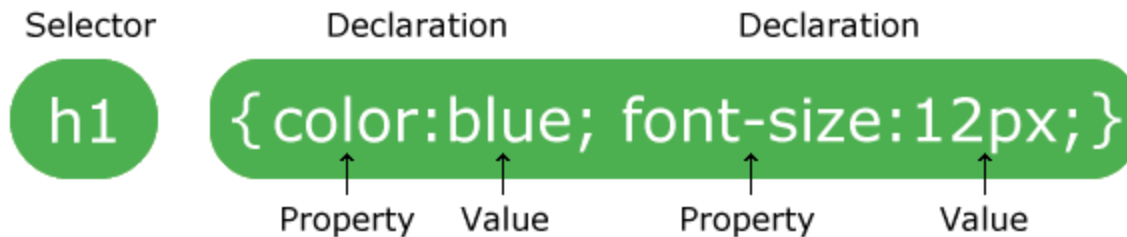
- Define styles for your web pages
- Variations in display for different devices and screen sizes.

CSS Saves a Lot of Work!

- The style definitions are normally saved in external .css files.
- With an external stylesheet file, you can change the look of an entire website by changing just one file!

CSS Syntax

- A CSS rule-set consists of a selector and a declaration block:



- **selector** points to the HTML element you want to style.
- **declaration** block contains one or more declarations separated by semicolons
Each declaration includes a CSS property name and a value, separated by a semi-colon

CSS Selectors

- CSS selectors are used to "find" (or select) HTML elements based on their element name, id, class, attribute, and more.
 - **The element Selector**
 - **The id Selector**
 - **The class Selector**
 - **Grouping Selectors**

I. The element Selector

- The element selector selects elements based on the element name.
- You can select all `<p>` elements on a page like this:

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

Example: Element Selector

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
p {
```

```
  color: red;
```

```
  text-align: center;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p>Hello World!</p>
```

```
<p>These paragraphs are styled with CSS.</p>
```

```
</body>
```

```
</html>
```

2. Id Selector

- The id selector uses the id attribute of an HTML element to select a specific element.
- **id of an element should be unique within a page**, so the id selector is used to select one unique element!
- To select an element with a specific id, write a hash (#) character, followed by the id of the element.
- The style rule below will be applied to the HTML element with id="para1":
- ```
#para1 {
 text-align: center;
 color: red;
}
```



# Example: Id Selector

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
#para1 {
```

```
 text-align: center;
```

```
 color: red;
```

```
}
```

```
</style></head>
```

```
<body>
```

```
<p id="para1">Hello World!</p>
```

```
<p>This paragraph is not affected by the style.</p>
```

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

# 3. Class Selector

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) character, followed by the name of the class.

- Example:

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

# Example: Class Selector

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
•center {
```

```
 text-align: center;
```

```
 color: red;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1 class="center">Red and center-aligned heading</h1>
```

```
<p class="center">Red and center-aligned paragraph.</p>
```

```
</body>
```

```
</html>
```

# 4. Grouping Selectors

- If you have elements with the same style definitions, like this:

```
h1 {
 text-align: center;
 color: red;
}
```

```
h2 {
 text-align: center;
 color: red;
}
```

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

# Grouping Selectors (Contd.)

- It will be better to group the selectors, to minimize the code.
- To group selectors, separate each selector with a comma.
- In the example below we have grouped the selectors from the code above:

```
h1, h2, p {
 text-align: center;
 color: red;
}
```

# Example: Group Selector

```
<!DOCTYPE html>
<html>
<head>
<style>
h1, h2, p {
 text-align: center;
 color: red;
}
</style>
</head>
<body>

<h1>Hello World!</h1>
<h2>Smaller heading!</h2>
<p>This is a paragraph.</p>
</body> </html>
```

# How to Apply CSS

- When a browser reads a style sheet, it will format the HTML document according to the information in the style sheet.
- Three Ways to Insert CSS
  - **External style sheet**
  - **Internal style sheet**
  - **Inline style**

# I. External Style Sheet

- The <link> element include a reference to the external style sheet goes inside the <head> section:

```
<! DOCTYPE html>
```

```
<html>
```

```
<head>
```

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

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

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



# I. External Style Sheet

**mystyle.css**

```
body {
 background-color: lightblue;
}
```

```
h1 {
 color: navy;
 margin-left: 20px;
}
```

## 2. Internal Style Sheet

- An internal style sheet may be used **if one single page has a unique style**
- Internal styles are defined within the `<style>` element, inside the `<head>` section of an HTML page

## 2. Internal Style Sheet (Example)

```
<!DOCTYPE html>
```

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

```
<style>
```

```
body {
```

```
 background-color: linen;
```

```
}
```

```
h1 {
```

```
 color: maroon;
```

```
 margin-left: 40px;
```

```
}
```

```
</style> </head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

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

# 3. Inline Styles

- An inline style may be used to apply a unique style for a single element.
- To use inline styles, add the style attribute to the relevant element.
- The style attribute can contain any CSS property.

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1 style="color:blue;margin-left:30px;">This is a
heading</h1>
```

```
<p>This is a paragraph.</p>
```

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

# CSS Background

- CSS background properties:
  - background-color
  - background-image
  - background-repeat
  - background-attachment
  - background-position

# I. Background Color

- The background-color property specifies the background color of an element.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
body {
```

```
background-color: lightblue;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>Hello World!</h1>
```

```
<p>This page has a light blue background color!</p>
```

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

# Example 2

```
<!DOCTYPE html>
```

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

```
<style>
```

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

```
div { background-color: lightblue; }
```

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

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>CSS background-color example!</h1>
```

```
<div>
```

This is a text inside a div element.

```
<p>This paragraph has its own background color.</p>
```

We are still in the div element.

```
</div>
```

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

## 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.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

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

```
</style> </head>
```

```
<body>
```

```
<h1>Hello World!</h1>
```

```
<p>This page has an image as the background!</p>
```

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



# Background Image - Repeat Horizontally or Vertically

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
body {
```

```
 background-image: url("gradient_bg.png");
```

```
 background-repeat: repeat-x;
```

```
}
```

```
</style> </head>
```

```
<body>
```

```
<h1>Hello World!</h1>
```

```
<p>Strange background image...</p>
```

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

# Background Image - Set position and no-repeat

```
<!DOCTYPE html>
<html> <head> <style>
body {
 background-image: url("img_tree.png");
 background-position: right top;
}
</style> </head>
<body>

<h1>Hello World!</h1>
<p>W3Schools background image example.</p>
<p>The background image is only showing once, but it is disturbing the
 reader!</p>

</body>
</html>
```

# CSS Border

**Specifies what kind of border to display.**

- dotted - Defines a dotted border
- dashed - Defines a dashed border
- solid - Defines a solid border
- double - Defines a double border
- groove - Defines a 3D grooved border. The effect depends on the border-color value
- ridge - Defines a 3D ridged border. The effect depends on the border-color value
- none - Defines no border
- hidden - Defines a hidden border

**The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border)**

# CSS Border

A dotted border.

A dashed border.

A solid border.

A double border.

A groove border.

A ridge border.

An inset border.

An outset border.

No border.

A hidden border.

A mixed border.

# Example (CSS border)

```
<!DOCTYPE html>
<html>
<head>
<style>
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {border-style: solid;}
</style>
</head>
<body>
<h2>The border-style Property</h2>
<p>This property specifies what kind of border to display:</p>
<p class="dotted">A dotted border.</p>
<p class="dashed">A dashed border.</p>
<p class="solid">A solid border.</p>
</body> </html>
```

# CSS Margins

The margin property is a shorthand property for the following individual margin properties:

- margin-top
- margin-right
- margin-bottom
- margin-left

# Example: (CSS Margins)

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> <style>
```

```
div {
```

```
border: 1px solid black;
```

```
margin: 25px 50px 75px 100px;
```

```
background-color: lightblue;
```

```
}
```

```
</style> </head>
```

```
<body>
```

```
<div>
```

This div element has a top margin of 25px, a right margin of 50px, a bottom margin of 75px, and a left margin of 100px.

```
</div>
```

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

# CSS padding

- The CSS padding properties are used to generate **space around an element's content, inside of any defined borders.**
- With CSS, you have full control over the padding.
- There are properties for setting the padding for each side of an element (top, right, bottom, and left).



# Example: (CSS padding)

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
div {
```

```
border: 1px solid black;
```

```
background-color: lightblue;
```

```
padding-top: 50px;
```

```
padding-right: 30px;
```

```
padding-bottom: 50px;
```

```
padding-left: 80px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<div>This div element has a top padding of 50px, a right padding of 30px, a bottom padding
of 50px, and a left padding of 80px.</div>
```

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

# CSS height and width

- The height and width properties are used to set the height and width of an element.
- The height and width can be set to auto (this is default.)
- Means that the browser calculates the height and width), or be specified in *length values*, like px, cm, etc., or in percent (%) of the containing block.

# Example: (CSS height and width)

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> <style>
```

```
p {
```

```
 height: 200px;
```

```
 width: 50%;
```

```
 background-color: powderblue;
```

```
}
```

```
</style> </head>
```

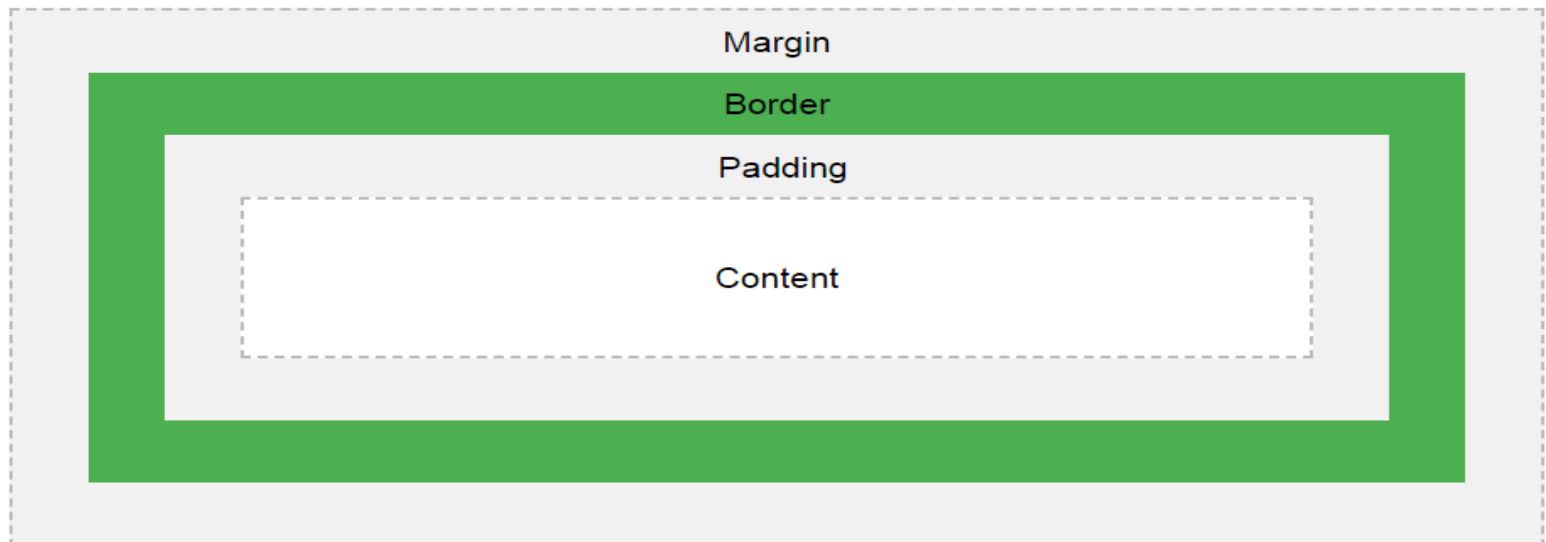
```
<body>
```

```
<p>This div element has a height of 200px and a width of
 50%:</p>
```

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

# 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 every HTML element. It consists of: margins, borders, padding, and the actual content. The image below illustrates the box model:



# CSS Text

## Text Color

- Set the color of the text.
- The color is specified by:
  - a color name - like "red"
  - a HEX value - like "#ff0000"
  - an RGB value - like "rgb(255,0,0)"

## Text Alignment

- Set the horizontal alignment of a text.
- A text can be left or right aligned, centered, or justified.

# CSS Text

## Text Decoration

- The text-decoration property is used to set or remove decorations from text.
- **Values specified:** overline, line-through, underline, none

## Text Transformation

- The text-transform property is used to specify uppercase and lowercase letters in a text.
- **Values specified:** uppercase, lowercase, capitalize

# Example: (CSS Text)

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
body {
```

```
 color: blue;
```

```
}
```

```
h1 {
```

```
 text-align: center;
```

```
}
```

```
h2 {
```

```
 text-decoration: line-through;
```

```
}
```

```
p.uppercase {
```

```
 text-transform: uppercase;
```

```
}
```

```
</style> </head>
```

# Example: (CSS Text)

```
<body>
```

```
<h1>This is heading 1</h1>
```

```
<h2>sample text decoration tag</h2>
```

```
<p>This is an ordinary paragraph. Notice that
this text is blue. The default text color for a
page is defined in the body selector.</p>
```

```
</body>
```

```
</html>
```



# CSS Fonts

## Font Family

- The font family of a text is set with the font-family property
- **Values specified:** Times New Roman, Georgia, Arial, Verdana etc

## Font Style

- Specify italic text.
- This property has three values:
  - **normal** - The text is shown normally
  - **italic** - The text is shown in italics
  - **oblique** - The text is "leaning" (oblique is very similar to italic, but less supported)

# CSS Fonts

## Font Size

- Sets the size of the text.
- The font-size value can be an absolute, or relative size.
- **Absolute size:**
  - Sets the text to a specified size
  - Does not allow a user to change the text size in all browsers (bad for accessibility reasons)
  - Absolute size is useful when the physical size of the output is known
- **Relative size:**
  - Sets the size relative to surrounding elements
  - Allows a user to change the text size in browsers

# CSS Fonts

## Font Weight

- Specifies the weight of a font
- **Values specified:** normal, bold, lighter

# Example:

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> <style>
```

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

```
p.italic {
 font-style: italic;
}
```

```
h1 {
 font-size: 40px;
}
```

## Example:

```
h2 {
 font-size: 2.5em; /* 40px / 16 = 2.5em
 */
}
```

```
p.normal {
 font-weight: normal;
}
```

```
</style>
```

```
</head>
```

# Example:

```
<html>
```

```
<body>
```

```
<h1>CSS font-family</h1>
```

```
<h2> font size in “em” units</h2>
```

```
<p class="serif">This is a paragraph, shown in the Times New Roman font.</p>
```

```
<p class="italic">This is a paragraph in italic style.</p>
```

```
<p class="normal">This is a paragraph.</p>
```

```
</body>
```

```
</html>
```

# CSS Links

With CSS, links can be styled in different ways.

Text Link

Text Link

Link Button

Link Button

```
<!DOCTYPE html>
<html> <head> <style>
a {
 color: hotpink;
}
</style> </head>
<body>
<p>This is a
link</p>
</body>
</html>
```

# CSS Links

- In addition, links can be styled differently depending on what **state** they are in.
- 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 moves over it
  - a:active - a link the moment it is clicked



# Example: (CSS Links)

```
<!DOCTYPE html>
```

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

```
/* unvisited link */
```

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

```
/* visited link */
```

```
a:visited {
 color: green;
}
```

```
/* mouse over link */
```

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

# Example: (CSS Links)

```
/* selected link */
```

```
a:active {
 color: blue;
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<p><a href="http://www.google.com"
 target="_blank">This is a link</p>
```

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

# CSS Lists

- 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
  - Add background colors to lists and list items

# Example: (CSS Lists)

```
<!DOCTYPE html>
<html>
<head>
<style>
ul.a {
 list-style-type: circle;
}

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

</style>
</head>
```

# Example: (CSS Lists)

```
<body>
```

```
<p>Example of unordered and ordered lists:</p>
```

```
<ul class="a">
```

```
Coffee
```

```
Tea
```

```
Coca Cola
```

```

```

```
<ol class="b">
```

```
Coffee
```

```
Tea
```

```
Coca Cola
```

```

```

```
</body>
```

```
</html>
```

# An Image as The List Item Marker

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

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

```
</style>
```

```
</head>
```

```
<body>
```

```

```

```
Coffee
```

```
Tea
```

```
Coca Cola
```

```

```

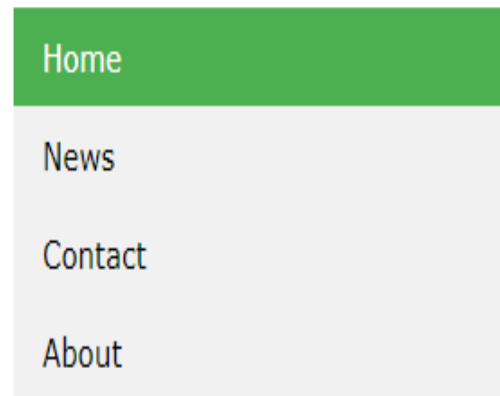
```
</body>
```

```
</html>
```

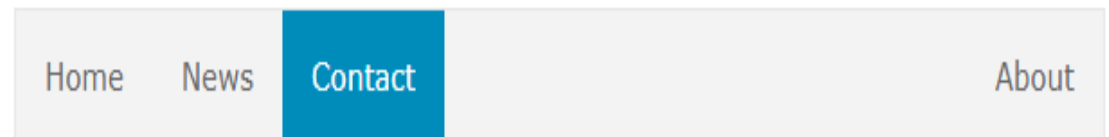
# CSS Navigation Bar

## Demo: Navigation Bars

Vertical



Horizontal



# Example: (Vertical Navigation Bar)

```
<html>
<head>
<style>
ul {
 list-style-type: none;
 margin: 0;
 padding: 0;
 width: 60px;
}

li a {
 display: block;
 background-color: #dddddd;
}
</style></head>
```



# Example: (Vertical Navigation Bar)

```
<body>
```

```

```

```
Home
```

```
News
```

```
Contact
```

```
About
```

```

```

```
</body>
```

```
</html>
```

A vertical navigation bar with a light gray background and a thin vertical line to its left. It contains four blue, underlined links stacked vertically: Home, News, Contact, and About.

[Home](#)  
[News](#)  
[Contact](#)  
[About](#)

# Example: (Horizontal Navigation Bar)

```
<!DOCTYPE html>
<html>
<head>
<style>
ul {
 list-style-type: none;
 margin: 0;
 padding: 0;
}

li {
 display: inline;
}
</style>
</head>
```

# Example: (Horizontal Navigation Bar)

```
<body>
```

```

```

```
Home
```

```
News
```

```
Contact
```

```
About
```

```

```

```
</body>
```

```
</html>
```

# Explore:

- CSS Drop Down Menu
- Image Gallery

# CSS Website Layout

A website is often divided into headers, menus, content and a footer:



# CSS Website Layout

## **Header**

- A header is usually located at the top of the website (or right below a top navigation menu). It often contains a logo or the website name

## **Navigation Bar**

- A navigation bar contains a list of links to help visitors navigating through your website:

# Content

- The layout in this section, often depends on the target users. The most common layout is one (or combining them) of the following:

**1-column** (often used for mobile browsers)

**2-column** (often used for tablets and laptops)

**3-column layout** (only used for desktops)



# CSS Website Layout

## **Footer**

- The footer is placed at the bottom of your page. It often contains information like copyright and contact info





# **CSS3 Features**

# CSS3 Borders

- The CSS3 provides two new properties for styling the borders of an element in a more elegant way —
  - border-image property for adding the images to borders
  - border-radius property for making the rounded corners without using any images.

# CSS border-image

```
<!DOCTYPE html>
```

```
<html>
```

```
<head> <style>
```

```
#borderimg {
```

```
 border: 10px solid transparent;
```

```
 padding: 15px;
```

```
 border-image: url(border.png) 30 round;
```

```
}
```

```
</style> </head>
```

```
<body>
```

```
<p id="borderimg">border-image: url(border.png) 30 round;</p>
```

```
<p>Here is the original image:</p>
```

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

border-image: url(border.png) 30 round;

Here is the original image:



# Creating CSS3 Rounded Corners

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Example of CSS3 Rounded Corners</title>
<style type="text/css">
 .box {
 width: 300px;
 height: 150px;
 background: #ffb6c1;
 border: 2px solid #f08080;
 border-radius: 20px;
 }
</style> </head>
<body>
 <div class="box"></div>
</body> </html>
```



# CSS3 Colors

- CSS3 adds some new functional notations for setting color values for the elements which are :
  - `rgba()`
  - `hsl()`
  - `hsla()`

# RGBA Color Values

- Colors defined in the RGBA model (red-green-blue-alpha) using the `rgba()` functional notation.
- RGBA color model are an extension of RGB color model with an alpha channel — which specifies the opacity of a color.
- The alpha parameter accepts a value from 0.0 (fully transparent) to 1.0 (fully opaque).
- **Example**

```
h1 {
 color: rgba(0,0,255,0.5);
}
p {
 background-color: rgba(0%,0%,100%,0.3);
}
```

# Example: RGBA Model

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-
color:rgba(255,0,0,0.3);}
#p2 {background-
color:rgba(0,255,0,0.3);}
#p3 {background-
color:rgba(0,0,255,0.3);}
#p4 {background-
color:rgba(192,192,192,0.3);}
#p5 {background-
color:rgba(255,255,0,0.3);}
#p6 {background-
color:rgba(255,0,255,0.3);}
</style>
</head>
```

```
<body>

<h1>Define Colors With RGBA
Values</h1>
<p id="p1">Red</p>
<p id="p2">Green</p>
<p id="p3">Blue</p>
<p id="p4">Grey</p>
<p id="p5">Yellow</p>
<p id="p6">Cerise</p>

</body> </html>
```

## Define Colors With RGBA Values

Red

Green

Blue

Grey

Yellow

Cerise

# HSL Color Values

- Colors also can be defined the HSL model (hue-saturation-lightness) using the `hsl()` functional notation.
- Hue is represented as an angle (from 0 to 360) of the color wheel or circle
- Saturation and lightness are represented as percentages.  
100% saturation means full color, and 0% is a shade of gray.
- 100% lightness is white, 0% lightness is black, and 50% lightness is normal.
- **Example**

```
h1 {
 color: hsl(360,70%,60%);
}
p {
 background-color: hsl(480,50%,80%);
}
```



# Example: HSL Model

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-color:hsl(120,100%,50%);}
#p2 {background-color:hsl(120,100%,75%);}
#p3 {background-color:hsl(120,100%,25%);}
#p4 {background-color:hsl(120,60%,70%);}
#p5 {background-color:hsl(290,100%,50%);}
#p6 {background-color:hsl(290,60%,70%);}
</style> </head>
<body>
<h1>Define Colors With HSL Values</h1>
<p id="p1">Green</p>
<p id="p2">Light green</p>
<p id="p3">Dark green</p>
<p id="p4">Pastel green</p>
<p id="p5">Violet</p>
<p id="p6">Pastel violet</p> </body>
</html>
```

## Define Colors With HSL Values

Green

Light green

Dark green

Pastel green

Violet

Pastel violet

# HSLA Color Values

- Colors can be defined in the HSLA model (hue-saturation-lightness-alpha) using the `hsla()` functional notation.
- HSLA color model are an extension of HSL color model with an alpha channel — which specifies the opacity of a color.
- The alpha parameter accepts a value from 0.0 (fully transparent) to 1.0 (fully opaque).
- **Example**

```
h1 {
 color: hsla(360,80%,50%,0.5);
}
p {
 background-color: hsla(480,60%,30%,0.3);
}
```

# Example: HSLA Model

```
<!DOCTYPE html>
<html>
<head>
<style>
#p1 {background-
color:hsla(120,100%,50%,0.3);}
#p2 {background-
color:hsla(120,100%,75%,0.3);}
#p3 {background-
color:hsla(120,100%,25%,0.3);}
#p4 {background-
color:hsla(120,60%,70%,0.3);}
#p5 {background-
color:hsla(290,100%,50%,0.3);}
#p6 {background-
color:hsla(290,60%,70%,0.3);}
</style>
</head>
```

```
<body>
<h1>Define Colors With HSLA
Values</h1>
<p id="p1">Green</p>
<p id="p2">Light green</p>
<p id="p3">Dark green</p>
<p id="p4">Pastel green</p>
<p id="p5">Violet</p>
<p id="p6">Pastel violet</p>
</body>
</html>
```

## Define Colors With HSLA Values

Green

Light green

Dark green

Pastel green

Violet

Pastel violet

# CSS3 Backgrounds

- The CSS3 provides several new properties to manipulate the background of an element like
  - background clipping
  - multiple backgrounds
  - option to adjust the background size.

# CSS3 background-size Property

- background-size property can be used to specify the size of the background images.
- Prior to CSS3, the size of the background images was determined by the actual size of the images.
- The background image size can be specified using the **pixels or percentage** values as well as the keywords **contain, and cover**.
- Negative values are not allowed.

# Example:

```
.box {
width: 250px;
height: 150px;
background: url("images/sky.jpg") no-repeat;
background-size: contain;
border: 6px solid #333;
}
```

**background-size: contain:**



**background-size: cover:**



# CSS3 background-clip Property

- The background-clip property can be used to specify whether an element's background extends into the border or not.
- The background-clip property can take the three values: **border-box, padding-box, content-box.**

## The background-origin Property

No background-origin (padding-box is default):

### Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetur 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.

background-origin: border-box:

### Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetur 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.

background-origin: content-box:

### Lorem Ipsum Dolor

Lorem ipsum dolor sit amet, consectetur 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.



# CSS3 Gradients

- The CSS3 gradient feature provides a flexible solution to generate **smooth transitions between two or more colors**.
- The elements with gradients can be scaled up or down to any extent without losing the quality, also the output will render much faster because it is generated by the browser.
- Gradients are available in two styles: *linear* and *radial*.

# Linear Gradient –Top to Bottom

```
/* Standard syntax */
background: linear-gradient(red, yellow);
}
```



# Linear Gradient - Left to Right

`/* Standard syntax */`

`background: linear-gradient(to right, red, yellow);`

`}`

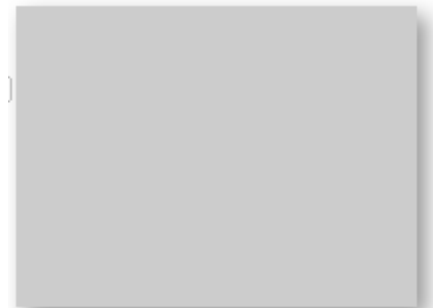


# CSS3 Drop Shadows

- Add drop shadow effects to the elements like you do in Photoshop without using any images.
- **Syntax: box-shadow: offset-x offset-y blur-radius color;**
  - **offset-x** — Sets the horizontal offset of the shadow.
  - **offset-y** — Sets the vertical offset of the shadow.
  - **blur-radius** — Sets the blur radius. The larger the value, the bigger the blur and more the shadow's edge will be blurred. Negative values are not allowed.
  - **color** — Sets the color of the shadow. If the color value is omitted or not specified, it takes the value of the color property.

# Example

```
.box{
width: 200px;
height: 150px;
background: #ccc;
box-shadow: 5px 5px 10px #999;
}
```



# CSS3 text-shadow Property

```
h1 {
```

```
text-shadow: 5px 5px 10px #666;
```

```
}
```

```
h2 {
```

```
text-shadow: 5px 5px 10px red, 10px 10px 20px
red;
```

```
}
```

**This is heading 1**

**This is heading 2**

# CSS3 2D Transforms

- With CSS3 2D transform feature you can perform basic transform manipulations such as **move, rotate, scale and skew** on elements in a two-dimensional space.
- A transformed element doesn't affect the surrounding elements, but can overlap them, just like the absolutely positioned elements.
- CSS3 transform property **uses the transform functions** to manipulate the coordinate system used by an element in order to apply the transformation effect.

# CSS3 2D Transforms:

## The translate() Function

- Moves the element from its **current position to a new position** along the X and Y axes.
- Syntax: translate(tx, ty).

```
<!DOCTYPE html>
```

```
<head> <meta charset="UTF-8">
```

```
<title>Example of CSS3 translate() Method</title>
```

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

```
img {
```

```
 transform: translate(200px, 50px); /* Standard syntax */
```

```
}
```

```
.box{
```

```
 margin: 50px;
```

```
 width: 153px;
```

```
 height: 103px;
```

```
 background: url("/examples/images/tortoise-transparent.png") no-repeat;
```

```
}</style> </head>
```



# Example: (contd...)

```
<body>
```

```
 <div class="box">
```

```

```

```
 </div>
```

```
</body>
```

```
</html>
```



# CSS3 2D Transforms:

## The rotate() Function

- Rotates the element around its origin by the specified angle.
- Syntax: **rotate(a).**

```
<!DOCTYPE html>
```

```
<head>
```

```
<title>Example of CSS3 rotate() Method</title>
```

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

```
 img {
```

```
 transform: rotate(30deg); /* Modern Browsers */
```

```
 }
```

```
 .box{
```

```
 margin: 50px;
```

```
 width: 120px;
```

```
 height: 110px;
```

```
 background: url("/examples/images/star-fish-transparent.png") no-repeat;
```

```
 }
```

```
</style>
```

```
</head>
```

# Example: (contd...)

```
<body>
```

```
 <div class="box">
```

```

```

```
</body>
```

```
</html>
```



Function	Description
<code>translate(tx,ty)</code>	Moves the element by the given amount along the X and Y-axis.
<code>translateX(tx)</code>	Moves the the element by the given amount along the X-axis.
<code>translateY(ty)</code>	Moves the the element by the given amount along the Y-axis.
<code>rotate(a)</code>	Rotates the element by the specified angle around the origin of the element, as defined by the <code>transform-origin</code> property.
<code>scale(sx,sy)</code>	Scale the width and height of the element up or down by the given amount. The function <code>scale(1,1)</code> has no effect.
<code>scaleX(sx)</code>	Scale the width of the element up or down by the given amount.
<code>scaleY(sy)</code>	Scale the height of the element up or down by the given amount.
<code>skew(ax,ay)</code>	Skews the element by the given angle along the X and Y-axis.
<code>skewX(ax)</code>	Skews the element by the given angle along the X-axis.
<code>skewY(ay)</code>	Skews the element by the given angle along the Y-axis.
<code>matrix(n,n,n,n,n,n)</code>	Specifies a 2D transformation in the form of a transformation matrix comprised of the six values.

# CSS3 3D Transforms

- With CSS3 3D transform feature you can perform basic transform manipulations such as **move, rotate, scale and skew on elements** in a three-dimensional space.
- The CSS3 transform property uses the **transform functions** to manipulate the coordinate system used by an element in order to apply the transformation effect.

# CSS3 3D Transforms:

## The translate3d() Function

- Moves the element from its current position to a new position along the X,Y and Z-axis.
- Syntax: **translate(tx, ty, tz).**

```
<!DOCTYPE html>
```

```
<head>
```

```
<title>Example of CSS3 translate3d() Method</title>
```

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

```
.container{
 width: 125px;
 height: 125px;
 perspective: 500px;
 border: 4px solid #e5a043;
 background: #fff2dd;
 margin: 30px;
}
```

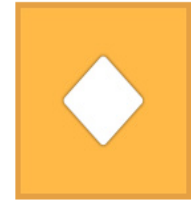
# Example: (contd...)

```
.transformed {
transform: translate3d(25px, 25px, 50px); /* Standard syntax */
}
</style>
</head>
<body>
 <h2>Before Translation:</h2>
 <div class="container">

 </div>
 <h2>After Translation:</h2>
 <div class="container">

 class="transformed" alt="Diamond Card">
 </div> </body> </html>
```

Before Translation:



After Translation:



# CSS3 3D Transforms:

## The rotate3d() Function

- The rotate3d() function rotates the element in 3D space by the specified angle around the [x,y,z] direction vector.
- Syntax: **rotate(vx, vy, vz, angle).**
- **Example:**

```
#myDiv {
 transform: rotateX(150deg);
}
```



# Example: 3D rotate method

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
div {
```

```
 width: 300px;
```

```
 height: 100px;
```

```
 background-color: yellow;
```

```
 border: 1px solid black;
```

```
}
```

```
#myDiv {
```

```
 transform: rotateX(150deg); /* Standard
 syntax */
```

```
}
```

```
</style> </head>
```

```
<body>
```

```
<h1>The rotateX() Method</h1>
```

```
<div>
```

```
This a normal div element.
```

```
</div>
```

```
<div id="myDiv">
```

```
This div element is rotated 150 degrees.
```


```
</div>
```

```
</body>
```

```
</html>
```

This a normal div element.

This div element is rotated 150 degrees.



Property	Description
<u>transform</u>	Applies a 2D or 3D transformation to an element
<u>transform-origin</u>	Allows you to change the position on transformed elements
<u>transform-style</u>	Specifies how nested elements are rendered in 3D space
<u>perspective</u>	Specifies the perspective on how 3D elements are viewed
<u>perspective-origin</u>	Specifies the bottom position of 3D elements
<u>backface-visibility</u>	Defines whether or not an element should be visible when not facing the screen

# CSS3 Transitions

- CSS transitions allows you to change property values smoothly, over a given duration.

Property	Description
<u><a href="#">transition</a></u>	A shorthand property for setting the four transition properties into a single property
<u><a href="#">transition-delay</a></u>	Specifies a delay (in seconds) for the transition effect
<u><a href="#">transition-duration</a></u>	Specifies how many seconds or milliseconds a transition effect takes to complete
<u><a href="#">transition-property</a></u>	Specifies the name of the CSS property the transition effect is for
<u><a href="#">transition-timing-function</a></u>	Specifies the speed curve of the transition effect

# Example: Transition

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<style>
```

```
div {
```

```
 width: 100px;
```

```
 height: 100px;
```

```
 background: red;
```

```
 transition: width 2s;
```

```
}
```

```
div:hover {
```

```
 width: 300px;
```

```
}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<h1>The transition Property</h1>
```

```
<div></div>
```

```
</body>
```

```
</html>
```



# CSS Animations

- CSS allows animation of HTML elements without using JavaScript or Flash!

Property	Description
<a href="#"><u>@keyframes</u></a>	Specifies the animation code
<a href="#"><u>animation</u></a>	A shorthand property for setting all the animation properties
<a href="#"><u>animation-delay</u></a>	Specifies a delay for the start of an animation
<a href="#"><u>animation-direction</u></a>	Specifies whether an animation should be played forwards, backwards or in alternate cycles
<a href="#"><u>animation-duration</u></a>	Specifies how long time an animation should take to complete one cycle
<a href="#"><u>animation-fill-mode</u></a>	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
<a href="#"><u>animation-iteration-count</u></a>	Specifies the number of times an animation should be played
<a href="#"><u>animation-name</u></a>	Specifies the name of the @keyframes animation
<a href="#"><u>animation-play-state</u></a>	Specifies whether the animation is running or paused
<a href="#"><u>animation-timing-function</u></a>	Specifies the speed curve of the animation

# Explore

- **CSS Tooltips**(extra information about something when the user moves the mouse pointer over an element)
- **CSS Object Fit**(<img> or <video> should be resized to fit its container)
- **CSS Buttons** (styling of buttons)
- **CSS Pagination** (move from one page to another <1>...<2>...