



**CSS 3**



# Table of Contents

---

Module	Topic
Module 1	CSS basics
Module 2	Selectors
Module 3	CSS box model
Module 4	Dimensions
Module 5	Positioning
Module 6	Fonts
Module 7	Text
Module 8	Color & Background

# Module 1: CSS basics

---

- ▶ Introduction
- ▶ How CSS works?
- ▶ Including CSS in web page

# Introduction to CSS

---

- ▶ CSS stands for Cascading Style Sheets.
- ▶ CSS is used to apply styles (look and formatting) to the content on our web page.
- ▶ CSS is a specification.
- ▶ CSS is a complementary language to HTML.

# How CSS works?

---

- ▶ CSS is a specification. Hence it is interpreted by the browser in its own way. Browsers are also free to write their own CSS properties.

# Including CSS in web page

---

- ▶ Using Inline Styles
- ▶ Using the <style> Element
- ▶ Using @import inside a <style> element
- ▶ Using the <link> Element

# Using Inline Styles

- ▶ In Inline style, CSS is contained inside an HTML attribute called 'style'.

```
<h1 style="color: red; background-color:#413;">Welcome</h1>
```

# Using the <style> Element

- ▶ The <style> element is included inside HTML & it contains CSS code.

```
<style>  
  header {  
    color: red;  
    background-color: #413;  
    font-size: 2.3em;  
  }  
</style>
```



# Using @import inside a <style> element

- ▶ You may write css into a separate .css file & include it inside HTML file using @import.

```
<style>  
    @import url(css/style.css);  
</style>
```

# Using the <link> Element

- ▶ The <link> element is placed inside <head>.
- ▶ This is the most recommended way of including css into HTML.

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

# Module 2: CSS Selectors

---

- ▶ What is a selector?
- ▶ Basic selectors
- ▶ Advanced selectors
- ▶ Pseudo-class
- ▶ Pseudo-element

# What is a selector?

- ▶ Selector is the part of a CSS rule set that actually selects the content you want to style. For example:

**.css file**

```
p { color: black; }
```

**.html file**

```
<p>This is black text.</p>
```

# Basic selectors

---

- ▶ Basic selectors set simple rules to choose contents for applying styles.
- ▶ Here are few basic selectors:
  - Universal selector
  - Element type selector
  - ID selector
  - Class selector
  - Attribute Selector
  - Multiple Selectors

# Universal selector

- ▶ The universal selector works like a wild card character (asterisk), selecting all elements on a page.
- ▶ For example using the following selector, all elements in html page will be green in color.

```
* { color: green; }
```

# Element type selector

- ▶ Element type selector matches one or more elements of the same name. In the following example, all `<ul>` elements in HTML page will set the given list-style & border.

```
ul {  
    list-style: none;  
    border: solid 1px #ccc;  
}
```

# ID selector

- ▶ An ID selector is declared using a hash symbol (#) preceding a string of characters also called as element ID.
- ▶ This selector matches HTML element having same ID. For example:

```
#container {  
    width: 960px;  
    margin: 0 auto;  
}
```



# Class selector

- ▶ Class selector is declared with a dot preceding a string of one or more characters.
- ▶ The class selector matches all elements on the page that have their class attribute set to the same value as the class.
- ▶ The class selector is the most useful of all CSS selectors.

```
.box {  
    padding: 20px;  
    margin: 10px;  
    width: 240px;  
}  
<div class="box"></div>
```

# Attribute selector

- ▶ The attribute selector targets elements based on the presence and/or value of HTML attributes, and is declared using square brackets.

```
input[type="text"] {  
    background-color: #444;  
    width: 200px;  
}
```

```
<input type="text">
```

# Multiple selectors

- ▶ We can combine multiple selectors together. For example we are combining class & element type selector together:

```
p, .box {  
    color: blue;  
}
```

- ▶ The above selector will set color to blue for all <p> elements & class as 'box'.

# Advanced selectors

---

- ▶ Advanced selectors give more control over setting custom rule for selecting content where styles will be applied.
- ▶ Advanced selectors are also called as 'combinators'.
- ▶ Here are the types of advanced selectors:
  - Descendant selector
  - Child selector
  - General Sibling selector
  - Adjacent Sibling selector

# Descendant selector

- ▶ Descendant selector allows you to limit the targeted elements to the ones who are descendants of another element. For example:

```
#container .box {  
    float: left; padding-bottom: 15px;  
}
```

- ▶ This declaration block will apply to all elements that have a class of box that are inside an element with an ID of container.

# Child selector

- ▶ Child selector is same as descendant selector, only difference is child selector is applied only on *'immediate children'*.
- ▶ Here we distinguish between parent & child selectors using greater than symbol (>).

```
#container > .box {  
    float: left;  
    padding-bottom: 15px;  
}
```

- ▶ Now the style will be applied to only immediate children of element id 'container'.

# General Sibling selector

- ▶ General sibling selector matches elements based on sibling relationship. This type selector is declared using tilde(~) character.

```
h2 ~ p {  
    margin-bottom: 20px;  
}
```

- ▶ Here all paragraph <p> elements those are having <h2> as sibling will be applied the style.

# Adjacent sibling selector

- ▶ Adjacent selector is applicable on only *'immediate sibling'* of the parent element. It is defined used plus (+) symbol.

```
p + p {  
    text-indent: 1.5em;  
    margin-bottom: 0;  
}
```

- ▶ This example will apply the specified styles only to paragraph elements that immediately follow other paragraph elements.



# Pseudo-class

- ▶ A pseudo-class is used to define a special state of an element.
- ▶ This special state may be a link is visited, activated, hovered etc.

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

```
a:hover { color: red; }  
a:active { color: blue; }  
div:hover { background-color: blue; }
```

# Pseudo-element

- ▶ 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::pseudo-element { property:value; }
```

- ▶ Here we use double colon (::) to distinguish between selector & pseudo element.

# Pseudo-element continue...

- ▶ The below pseudo element will apply the style on first line of every paragraph `<p>` element.

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

# CSS Pseudo elements

Selector	Example	Example description
<u>::after</u>	p::after	Insert something after the content of each <p> element
<u>::before</u>	p::before	Insert something before the content of each <p> element
<u>::first-letter</u>	p::first-letter	Selects the first letter of each <p> element
<u>::first-line</u>	p::first-line	Selects the first line of each <p> element
<u>::selection</u>	p::selection	Selects the portion of an element that is selected by a user

# CSS rules overriding

---

- ▶ Any inline style sheet takes highest priority. So, it will override any rule defined in `<style>...</style>` tags or rules defined in any external style sheet file.
- ▶ Any rule defined in `<style>...</style>` tags will override rules defined in any external style sheet file.
- ▶ Any rule defined in external style sheet file takes lowest priority, and rules defined in this file will be applied only when above two rules are not applicable.



# Module 3: CSS box model

---

- ▶ What is a box model?
- ▶ Box model components
- ▶ States of HTML elements
- ▶ Margin
- ▶ Borders
- ▶ Padding
- ▶ Outlining
- ▶ Visibility & Display

# CSS box model

- ▶ The box model refers to the (usually) invisible rectangular area that is created for each HTML element.



# Box model components

---

## ► Content:

- Content is a visible part of HTML element that you want to show to end user.

## ► Padding:

- The padding is the space around the content. It can be defined for an individual side i.e. padding-left, padding-right, padding-top, padding-bottom.

## ► Border:

- The border of an element is defined using the border property. This is a shorthand property that defines the element's border-width, border-style, and border-color. For example, border: 4px dashed orange.



# Box model components

---

## ► Margin:

- Margin is similar to padding except unlike padding, the margin portion of an element exists outside the element. A margin creates space between the targeted element and surrounding elements.

# States of HTML elements

---

- ▶ Every HTML element has three different states:
  - Invisible state
  - Block state
  - Inline state

# Invisible state

---

- ▶ Invisible state elements are not supposed to be rendered by browser.
- ▶ For example `<style>`, `<meta>` tags etc.

# Block state

---

- ▶ A block-level element is more of a structural, layout related element.
- ▶ A block level element will naturally span the entire available width, without concern about how much horizontal space it actually needs.
- ▶ As a result of that, a block level element will automatically push following content to a new line.
- ▶ Thus block level element can be seen as a paragraph.
- ▶ Block-level elements include elements like `<div>`, `<p>`, `<section>` etc.

# Inline state

---

- ▶ An inline element only takes up the space it needs to render its content and after that, more inline elements can be displayed.
- ▶ The width and height properties are ignored for inline elements. Instead, you can use the line-height property to make an inline element smaller or bigger, as the space between each line grows or shrinks.
- ▶ Inline elements include elements like `<span>`, `<b>`, and `<em>` etc.

# Margin

- ▶ Margin is an outer, invisible border around your element.
- ▶ The default value for the margin properties is auto, which usually translates to zero.

```
<style type="text/css">
.box {
  background-color: DarkSeaGreen;
  width: 100px;
  height: 100px;
  margin-top: 10px;
  margin-right: 5px;
  margin-bottom: 10px;
  margin-left: 5px;
}
</style>
```

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

# Margin with shorthand

- ▶ You can also specify only 'margin' property instead of specific properties like margin-top, margin-right etc. This is called as shorthand properties.

```
<div class="box" style="margin: 10px 10px 10px 10px;">
```

***Sets top, right, bottom & left margin as 10px.***

```
<div class="box" style="margin: 10px 20px 10px;">
```

***Sets top & bottom margin as 10px & left/right as 20px.***

```
<div class="box" style="margin: 10px 20px;">
```

***Sets top/bottom margin as 10px & left/right as 20px.***

```
<div class="box" style="margin: 10px;">
```

***Sets all margins top/bottom/right/left as 10px.***

# Relative Margins using 'em'

- ▶ CSS allows us to define relative size unit using '*em*'.
- ▶ A single 'em' is always equal to whatever the value is of the font-size property on the element to which the em unit is applied.

```
.box {  
    margin: 1em;  
}  
  
<div class="box" style="font-size: 1em;">Box</div>  
  
<div class="box" style="font-size: 2em;">Box 2</div>  
  
<div class="box" style="font-size: 3em;">Box 3</div>
```



# Relative Margins using 'rem'

- ▶ Using 'em' unit sometime becomes complicated when their value is inherited by parent element.

```
html { font-size: 22px; }  
.box { font-size: 20px; padding: 1.5em; }  
p {  
    font-size: 14px;  
    padding: 2rem;  
}
```

- ▶ Instead of calculating their value based on the element's font-size value, rem units calculate their value based on the font-size value set on the root element (hence "rem," or "root em").

# Negative margins

- ▶ Margin can be declared as negative value as well. It is used to negate the effect of padding.

```
.box-header {  
    margin: -10px -10px 10px -10px;  
    padding: 5px 10px;  
}
```

# Borders

- ▶ HTML elements doesn't have a border by default.
- ▶ You can give border to any element as follows:

```
.box {  
    width: 100px;  
    height: 100px;  
    border-color: red;  
    border-width: 2px;  
    border-style: solid;  
}  
<div class="box">Hello, world!</div>
```

- ▶ The border-width can be absolute or relative value.  
Border style can be one of hidden, dotted, dashed, solid, double, groove, ridge, inset and outset.

# Border with shorthand

```
.box {  
    border: 2px solid blue;  
}
```

- ▶ You can mention border top, right, bottom & left properties using shorthand as shown above. You can combine style, color & width as single property as follows:

```
.box {  
    width: 100px;  
    height: 100px;  
    border-style: solid dashed ridge dotted;  
    border-color: red green blue orange;  
    border-width: 1px 2px 3px 4px;  
}
```

# Border with radius

```
.box {  
  border: 3px solid blue;  
  border-radius: 5px;  
}
```

Hello, world!

# Padding

- ▶ Padding is an inner, invisible border around HTML element.

```
.box {  
    padding-top: 5px;  
    padding-right: 10px;  
    padding-bottom: 5px;  
    padding-left: 10px;  
}
```

```
<div class="box" style="padding: 10px 10px 10px 10px;">
```

***Sets top, right, bottom & left padding as 10px.***

```
<div class="box" style="padding: 10px 20px 10px;">
```

***Sets top & bottom padding as 10px & left/right as 20px.***

```
<div class="box" style="padding: 10px 20px;">
```

***Sets top/bottom padding as 10px & left/right as 20px.***

```
<div class="box" style="padding: 10px;">
```

***Sets all padding top/bottom/right/left as 10px.***

# Relative Padding

- ▶ You can provide relative padding using 'em' units.

```
.box {  
    padding: 1em;  
}  
<div class="box" style="font-size: 2em;">
```



# Outlining

- ▶ Outline is an extra border for HTML element.

```
.box {  
    outline-width: 3px;  
    outline-style: solid;  
    outline-color: pink;  
}
```

- ▶ You cannot apply a different outline width, style and color for the four sides of an element, like you can with the border.
- ▶ The outline is not a part of the element's dimensions, like the border.



Hello, world!

# Outlining with shorthand

```
.box {  
  outline: 3px solid red;  
}
```



Hello, world!

# Visibility property

---

- ▶ Sometimes you want to show or hide an element depending upon a condition. This is possible using 'visibility'.

```
<div class="box" style="visibility: hidden;">Box 2</div>
```

- ▶ Note that making an element hidden still reserves space for the element.



# Display property

---

- ▶ Display element will control the existence of an element in the view.

```
<div class="box" style="display: none;">Box 2</div>
```

- ▶ Display property can have 3 possible values: none, inline or block.

# Module 4: CSS Dimensions

---

- ▶ Width & height.
- ▶ Minimum width/ height.
- ▶ Overflow

# Width & height

- ▶ We can apply width & height properties to only block level elements like <div>, <p>, <section> etc. This is because block elements use entire available horizontal space & only required vertical space.
- ▶ Inline elements like <span> etc. ignore width & height.

```
.box {    width: auto;    height: auto;    } //Auto adjust
```

```
.box {    width: 100px;    height: 200px;    } //Absolute values
```

```
.box {    width: 25%;    height: 30%;    } //Relative values
```

# Minimum & maximum width & height

```
.box { width: 100px; height: 200px; }
```

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

```
<div class="box" style="min-height: 300px; min-width: 200px;">
```

```
<div class="box" style="max-height: 300px; max-width: 200px;">
```

# 'overflow' property

Sometimes an element has larger contents than its dimensions. In such case, 'overflow' property helps browser to display hidden contents or not. If you wish to scroll in case larger contents then use overflow as 'scroll'.

```
<div class="box" style="overflow: visible;">
```

Box 1 - A  
box with  
many,  
many  
words in it,  
designed  
to cause



# 'overflow' property values

---

'overflow' property can have four possible values:

1. **Visible** (default): It expands the contents beyond the border.
2. **Hidden**: Contents going beyond border are kept hidden.
3. **Auto**: It leaves the problem up to the browser to handle.
4. **Scroll**: Provides scroll bars to view overflow contents.

# Module 5: CSS Positioning

---

- ▶ Relative positioning
- ▶ Absolute positioning
- ▶ Fixed positioning
- ▶ Floating positioning

# Positioning of elements

---

- ▶ Positioning elements in web page is very important aspect of css. Correct positioning makes web page impressive & attractive.
- ▶ Positioning is mentioned using 'position' property which can have following values:
  1. Static
  2. Relative
  3. Absolute
  4. Fixed
  5. Floating

# Static position

- ▶ In 'static' positioning, elements are placed from top to bottom, within the available space of their parent.
- ▶ 'static' is a default positioning.

```
#inner-div { position: static }
```

```
<div id='outer-div'>
```

```
    <div id='inner-div'>XXX</div>
```

```
</div>
```

# Relative position

- ▶ In 'relative' positioning, elements are placed relative to its parent element.
- ▶ 'static' is a default positioning.

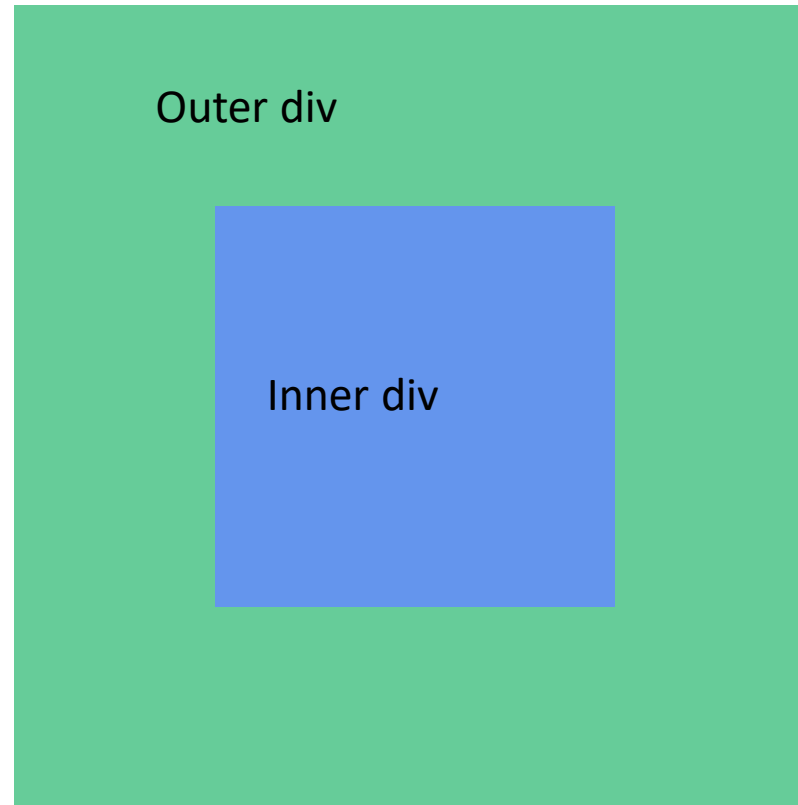
```
#outer-div { width: 100px, height: 100px }
```

```
#inner-div { width: 50px, height: 50px, top: 100px, left: 100px,  
position: relative }
```

```
<div id='outer-div'>  
  <div id='inner-div'>XXX</div>  
</div>
```

# Relative position continue...

---



# Absolute position

- ▶ Using absolute positioning, you can position your element at any location with respect to browser window or parent element.
- ▶ Absolute positioning comes with few properties like top, bottom, left & right.

```
#absolute-element {  
    position: absolute;  
    width: 100px;    height: 100px;  
    top: 35px;    left: 35px;  
}  
<div id='absolute-element'></div>
```

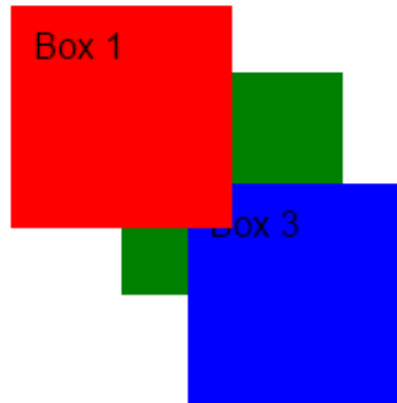
# z-index property

- In case, elements overlapped then which element should be shown top is controlled by a property called 'z-index'.

```
<div class="box" style="background-color: red; top: 10px; left: 10px; z-index: 3;">Box 1</div>
```

```
<div class="box" style="background-color: green; top: 60px; left: 60px; z-index: 1;">Box 2</div>
```

```
<div class="box" style="background-color: blue; top: 100px; left: 100px; z-index: 2;">Box 3</div>
```

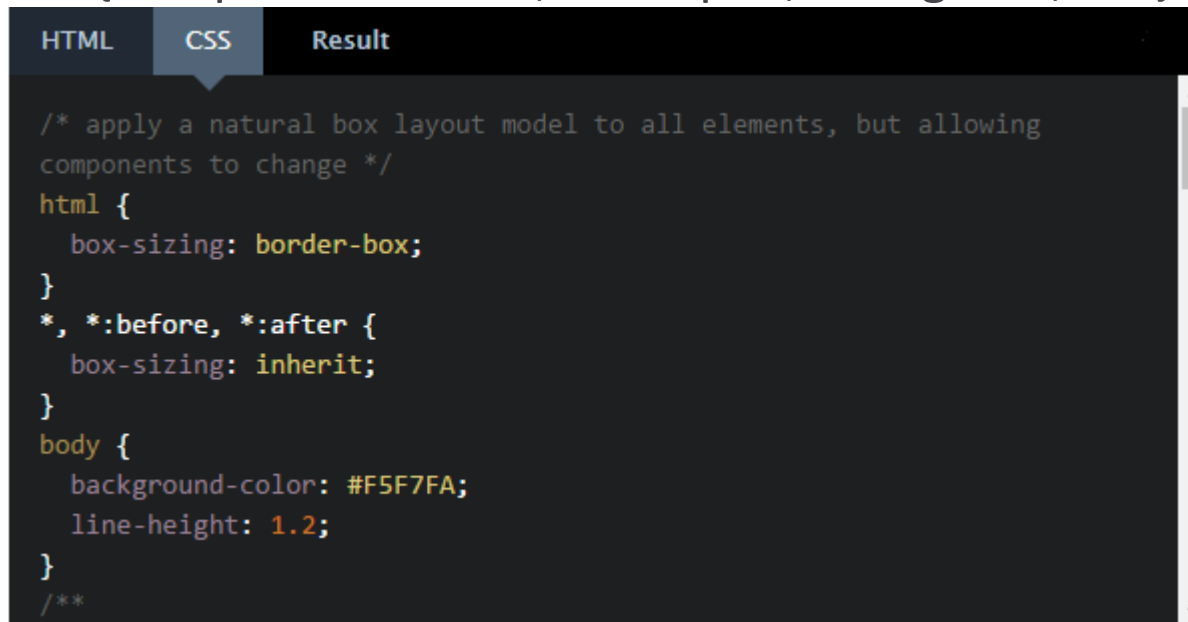




# Fixed position

- Fixed position is same like absolute. Only in case of scroll up or down the position of fixed element remains at the designated place.

```
.menu-fixed { position: fixed; top: 0; right: 0; }
```

A screenshot of a web browser's developer tools, specifically the CSS pane. The 'HTML' tab is selected, showing a list of elements. The 'CSS' tab is active, displaying a list of CSS rules. The 'Result' tab is also visible. The CSS rule for '.menu-fixed' is highlighted, showing 'position: fixed; top: 0; right: 0;'. The background of the browser shows a dark theme with a blue header and a white footer.

```
/* apply a natural box layout model to all elements, but allowing
components to change */
html {
  box-sizing: border-box;
}
*, *:before, *:after {
  box-sizing: inherit;
}
body {
  background-color: #F5F7FA;
  line-height: 1.2;
}
/**
```

# 'float' property

- ▶ The 'float' property allows us to take a block level element out of the normal order and let other elements float around it.

```
.image { float: left; }  
  
<div class="container">  
      
    Lorem ipsum dolor sit amet....  
</div>
```

# Module 6: Fonts

---

- ▶ Types of fonts
- ▶ Applying fonts
- ▶ Font weight
- ▶ Font style
- ▶ Font size

# Introduction to fonts

---

- ▶ Web page is a media of sharing information with users. A web page involves text, images, videos etc. However, the most important component is the text because most of web page information is shared using text only.
- ▶ Optimizing typography of web page increases its accessibility, readability and usability. Hence we need to learn how to style our text.

# Serif & Sans-serif fonts

- ▶ There are basically two types of fonts Serif & Sans-serif. For example:

`<p style="font-family:serif;">This is your browsers default serif font</p>`

`<p style="font-family:sans-serif;">This is your browsers default sans-serif font</p>`

- ▶ Serif fonts are small decorative flourish at the end where as Sans-serif does not flourish.

F

Sans-serif

F

Serif

# Serif fonts

American Typewriter

Bookman

Clearface

Didot

**Elephant**

Footlight

Georgia

Hightower Text

Imprint

*Janson*

Korinna

Lexicon

MS Serif

New York

Times New Roman

Utopia

# Sans-serif fonts

Agency FB

Arial

BANK GOTHIC

Calibri

Dotum

**Eurocrat**

**Folio**

Gotham

**Helvetica**

**Kabel**

**Industria**

**JOHNSTON**

Roboto

Lucida Grande

# Applying fonts

- ▶ `<p style="font-family: Impact, sans-serif; ">Impact - a bold font</p>`
- ▶ `<p style="font-family: Century Gothic, sans-serif; ">Century Gothic, a personal favorite</p>`
- ▶ `<p style="font-family: Times New Roman, Times, serif; ">Times New Roman - the classic choice</p>`

**Impact - a bold font**

Century Gothic, a personal favorite

Times New Roman - the classic choice



# Font weight

- ▶ The font-weight property defines how bold your text is. The possible font weight values are:
- ▶ normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900, and inherit.

```
<style>
    .weight200 {font-weight:200 }
    .weight800 {font-weight:800 }
</style>
<p class="weight200">Text with 200
weight</p>
<p class="weight800"> Text with 800 weight
</p>
```

# Font style

- ▶ The font-style property defines whether or not your text is italic or oblique.
- ▶ `p.normal { font-style: normal; }`
  - *This is a paragraph, normal.*
- ▶ `p.italic { font-style: italic; }`
  - *This is a paragraph, italic.*
- ▶ `p.oblique { font-style: oblique; }`
  - *This is a paragraph, oblique.*

# Italic vs Oblique font style

- ▶ The italic font style is decorative where as oblique font style is a simple slanted version of normal font.

ITC Legacy Serif Book	Univers Light
<i>ITC Legacy Serif Book</i> <b>Italic</b>	<i>Univers Light</i> <b>Oblique</b>

# Font size

- ▶ Font size can be mentioned either in 'em' or in 'percentage'.

```
.textLarge { font-size: 2em; }
```

```
.textRegular{ font-size:1em; }
```

```
body { font-size:100%; }
```

```
p { font-size: 130% }
```

```
.smallerText { font-size: 50%; }
```

# Module 7: Text

---

- ▶ Text alignment
- ▶ Text spacing
- ▶ Text transformation
- ▶ Text decoration
- ▶ Text indentation
- ▶ Text alignment



# Text alignment

---

We can use 'text-align' property to specify the alignment.

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

The 'text-align' can have following options:

1. left (default)
2. right
3. center
4. justify

# Word spacing

- ▶ Word spacing features allows us to define custom space between two words.
- ▶ Word spacing reduces text readability.
- ▶ Word spacing can be used for headings or block quotes.

```
h2 { word-spacing: 2.2em; }
```

```
<h2>This is extra word-spacing</h2>
```

**This is extra word-spacing**

# Letter spacing

- ▶ The letter-spacing property lets you to control the space between two letters.

```
h2 { letter-spacing: 0.25em; }  
<h2>Extra space between letters</h2>
```

**E x t r a s p a c e b e t w e e n l e t t e r s**



# Text transformation

- Using text transformation properties you can transform your text to uppercase, lowercase, or capitalize.

```
p { text-transform: uppercase; }
```

```
<p style="text-transform: uppercase;">This is my  
uppercase text.</p>
```

THIS IS MY UPPERCASE TEXT.

# Text decoration

- Using text-decoration property, we can decorate our text. For example we can give underline, line through or overline to the text.

```
<div style="text-decoration: underline;">Underline text</div>
```

```
<div style="text-decoration: line-through;">Line through  
text</div>
```

```
<div style="text-decoration: overline;">Overline text</div>
```

Underline text

~~Line through text~~

Overline text

# Text indent

- Using text-indent property, we can set the offset of starting first line of text.

```
p { text-indent: 100px; }  
p { text-indent: 2em; }  
p { text-indent: 10%; }
```

Lorem ipsum dolor  
sit amet, consectetur  
adipiscing elit. Sed  
condimentum augue sed  
diam semper rhoncus.  
Curabitur porttitor mattis  
tortor, eget aliquam lectus  
porta ac.

# Text alignment

- ▶ 'text-align' property allows us to specify the text alignment.

```
<p style="text-align: left;">Left aligned text</p>
```

```
<p style="text-align: right;">Right aligned text</p>
```

```
<p style="text-align: center;">Center aligned text</p>
```

```
<p style="text-align: justify;">Justify aligned text</p>
```

Lorem ipsum dolor sit amet,  
consectetur adipiscing elit.  
Suspendisse a lectus  
mattis, consequat mi vitae,  
tristique ipsum. In hac  
habitasse platea dictumst.  
Integer sit amet aliquet  
dolor.

# Module 8: Color & Background

---

- ▶ Applying color
- ▶ Background image
- ▶ CSS transitions

# Text color & background color

- ▶ Text color & background color can be set using 'color' & 'background-color' properties respectively.

```
<p style="color: Red;">Red text</p>
```

```
<p style="background-color: Cyan; color: Red;">Red  
text</p>
```

Red text

Red text

# Different ways to set color

---

Color can be set in different ways as follows:

```
<p style="color: Red;">Red text</p>
```

```
<p style="color: rgb(255,0,0);">Red text</p>
```

```
<p style="color: #ff0000;">Red text</p>
```

Refer to the color guide at <http://www.cssportal.com/css3-color-names/>

# Background image

Apart from background color, we can also set background as image using 'background-image' property.

```
.box { background-image: 'blue_background.png';}
```

```
<div class="box">Welcome to CSS</div>
```



**Welcome to  
CSS**



# Repeating background image

---

The size of your container can be bigger than image size. In such cases CSS by default repeats the image so that whole container's background will be occupied by the image. CSS also provides us flexibility to control the image repeats using 'background-repeat' property with possible values:

- repeat (default)
- repeat-x
- repeat-y
- no-repeat

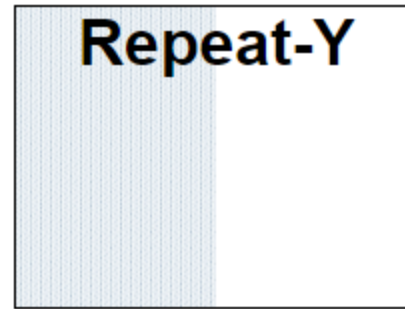
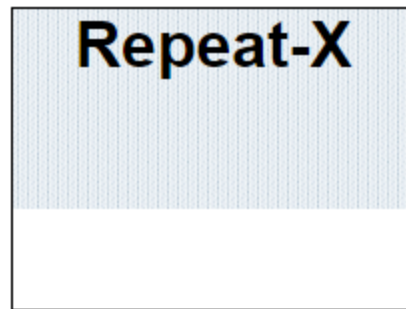
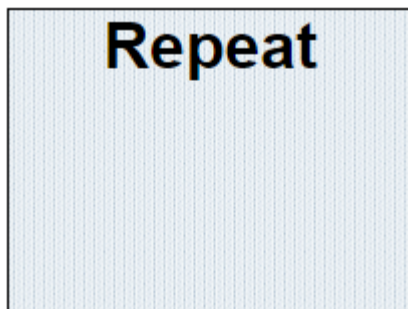
# Repeating background image example

`<div class="box" style="background-repeat: repeat;">Repeat</div>`

`<div class="box" style="background-repeat: repeat-x;">Repeat-X</div>`

`<div class="box" style="background-repeat: repeat-y;">Repeat-Y</div>`

`<div class="box" style="background-repeat: no-repeat;">No-Repeat</div>`



# Background image positioning

We can specify position of background image using 'background-position' property. Here are few possible combinations of background positioning:

```
<div class="box" style="background-position: top left;">
```

```
<div class="box" style="background-position: top right;">
```

```
<div class="box" style="background-position: bottom right;">
```

```
<div class="box" style="background-position: 20% 50%;">
```

```
<div class="box" style="background-position: 20px 50px;">
```

```
<div class="box" style="background-position: top 50%;">
```

# CSS Layouts



- ▶ CSS page layout techniques allow us to take elements contained in a web page and control where they are positioned relative to their default position in normal layout flow, the other elements around them, their parent container, or the main viewport/window.
  - Normal flow
  - The display property
  - Flexbox
  - Grid
  - Floats
  - Positioning
  - Table layout
  - Multiple-column layout



# Normal flow

- ▶ Normal flow is how the browser lays out HTML pages by default when you do nothing to control page layout.

```
<p>I love my cat.</p>  
  <ul>  
    <li>Buy cat food</li>  
    <li>Exercise</li>  
    <li>Cheer up friend</li>  
  </ul>  
<p>The end!</p>
```

# The display property

---

- ▶ The main methods of achieving page layout in CSS are all values of the display property. This property allows us to change the default way something displays.
- ▶ For example, the `<li>` element is `display: block` by default, meaning that list items display one below the other in our English document. If we change the display value to `inline` they now display next to each other, as words would do in a sentence.
- ▶ The two values most important for our purposes when discussing layout are **display: flex** and **display: grid**.



# Flexible Box Layout

---

- ▶ Designed to make it easy for us to lay things out in one dimension either as a row or as a column.
- ▶ To use flexbox, apply **display: flex** to the parent element of the elements you want to lay out; all its direct children then become flex items.



# Grid Layout

---

- ▶ While flexbox is designed for one-dimensional layout, Grid Layout is designed for two dimensions — lining things up in rows and columns.
- ▶ You can switch on Grid Layout with a specific value of display — `display: grid`.
- ▶ In addition to using **display: grid**, we are also defining some row and column tracks on the parent using the **grid-template-rows** and **grid-template-columns** properties respectively.
- ▶ The start and end line of each item can be specified using the **grid-column** and **grid-row** properties. This causes the items to span multiple tracks.





# Table layout

---

- ▶ Web developers use tables for entire web page layouts — putting headers, footers, different columns, etc. in various table rows and columns.
- ▶ **display: table-caption;** — which makes it act like a table <caption>



# Multi-column layout

---

- ▶ The multi-column layout module gives us a way to lay out content in columns, similar to how text flows in a newspaper.
- ▶ To turn a block into a multicolumn container we use either the **column-count** property, which tells the browser how many columns we would like to have, or the **column-width** property, which tells the browser to fill the container with as many columns of at least that width.



# CSS transitions

---

- ▶ CSS transitions allows to change property values smoothly (from one value to another), over a given duration. It includes total four properties:
  - **transition-property:** It specifies which property the transition is applied to - be it color, background, border, and so. Possible values are none, all, property, initial & inherit.
  - **transition-duration:** It specifies how many second or milliseconds a transition takes to complete.

# CSS transitions continue...

---

- **transition-timing-function:** With the transition timing function, you can change the speed-curve of the transition effect. Possible values are linear, ease-in, ease-out, ease-in-out, cubic-bezier.
- **transition-delay:** The transition-delay property specifies when the transition effect will start. Possible values are time, initial & inherit.



## US – CORPORATE HEADQUARTERS

1248 Reamwood Avenue,  
Sunnyvale, CA 94089  
Phone: (408) 743 4400

343 Thornall St 720  
Edison, NJ 08837  
Phone: (732) 395 6900

## UK

57 Rathbone Place,  
4th Floor, Holden House,  
London, W1T 1JU , UK

89 Worship Street Shoreditch,  
London EC2A 2BF, UK  
Phone: (44) 2079 938 955

## INDIA

### Mumbai

4th Floor, Nomura  
Powai , Mumbai 400 076  
Phone: +91 (22) 3051 1000

### Pune

5th floor, Amar Paradigm Baner Road  
Baner, Pune 411 045  
Phone: +91 (20) 6604 6000

### Bangalore

4th Floor, Kabra Excelsior,  
80 Feet Main Road, Koramangala 1st Block,  
Bengaluru (Bangalore) 560034  
Phone: +91 (80) 4666 1666