

HTML

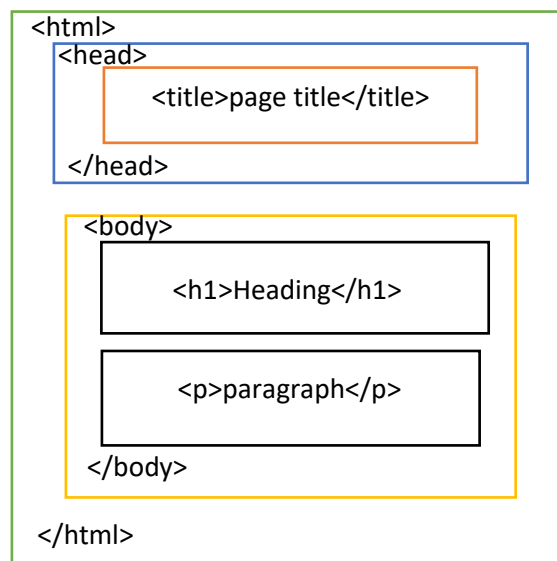
What is HTML?

HTML stands for **Hyper Text Markup Language**

It is a language which is used to design Webpages.

In HTML, Every Element contains a start tag (<>) and an end tag(</>)

HTML Page Structure:



- The <head> element is a container for metadata (data about data) and is placed between the <html> tag and the <body> tag.

The <title> element:

- defines a title in the browser toolbar
- provides a title for the page when it is added to favorites
- displays a title for the page in search engine-results

EXAMPLE of Simple HTML:

```
<!--Double Slash(//) is also used to write comments. -->
<!DOCTYPE html>
<!-- This is a declaration that tells it is an HTML page -->
<html>
  <!-- It is the main element in the Html page -->
<head>
  <title>Your Own Title of Your Webpage</title>
  <!-- Specifies title of the HTML page -->
```

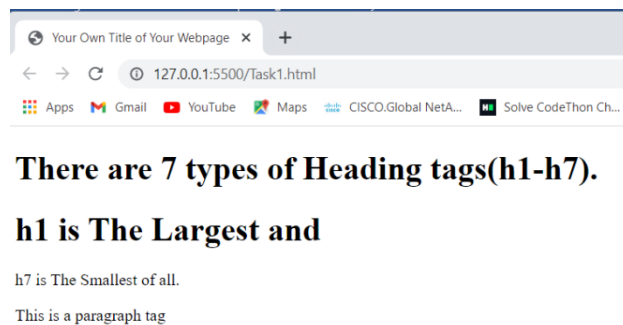
```

</head>

<body>
<!-- It consists of all contents that will appear in our desired webpage. -->
  <h1>There are 7 types of Heading tags(h1-h7).</h1>
  <h1>h1 is The Largest and</h1>
  <h7> h7 is The Smallest of all </h7>
  <p>This is a paragraph tag</p>
</body>
</html>

```

The Output:



HTML Layout Elements:

HTML has several semantic elements that define the different parts of a web page:

<header>	• <header> - Defines a header for a document or a section
<nav>	• <nav> - Defines a set of navigation links
<section>	• <section> - Defines a section in a document
<article>	• <article> - Defines an independent, self-contained content
<aside>	• <aside> - Defines content aside from the content (like a sidebar)
<footer>	• <footer> - Defines a footer for a document or a section
	• <details> - Defines additional details that the user can open and close on demand
	• <summary> - Defines a heading for the <details> element

Points to Remember:

- In HTML, spaces and new lines are ignored.
-
 is a break tag which is used to give a break between two elements.it doesn't have an end tag.
- HTML tags are not case sensitive: <P> means the same as <p>.
- The <hr> tag defines a thematic break in an HTML page

HTML Text-Formatting :

 - Bold text, without any extra importance, displayed in bold

 - Important text, text with strong importance, displayed in bold.

<i> - Italic text, content inside is typically displayed in italic which is used to represent some words alternatively.

 - Emphasized text, is typically displayed in italic and are pronounced with verbal stress.

<mark> - Marked text, it is used to mark or highlight.

<small> - Smaller text , displays text in small size.

 - Deleted text, it is used to strike a line.

<ins> - Inserted text, it is used to underline inserted text.

<sub> - Subscript text, the text appears half a character below the normal line.

<sup> - Superscript text, the text appears half a character above the normal line.

Example:

```
<!DOCTYPE html>
<html>
<body>
<b>Lata Mangeshkar</b> (born as <small> Hema Mangeshkar</small>;<strong> 28
September 1929 –6 February 2022</strong>) was <i>an Indian playback singer</i>
and <em>occasional music composer.</em>She is widely considered to have been
<mark>one of the greatest and most influential singers in
India</mark>.<del>His</del> <ins>Her</ins> contribution to the <sub>Indian
music industry </sub>in a career spanning seven decades gained her honorific
titles such as<sup> the Nightingale of India, Voice of the Millennium and
Queen of Melody</sup>
</body>
</html>
```

Output:

Lata Mangeshkar (born as Hema Mangeshkar; **28 September 1929 – 6 February 2022**) was *an Indian playback singer* and *occasional music composer*. She is widely considered to have been **one of the greatest and most influential singers in India**.~~His~~ Her contribution to the Indian music industry in a career spanning seven decades gained her honorific titles such as ^{the Nightingale of India, Voice of the Millennium and Queen of Melody}

Just Know :

- <address> Tag: This tag defines the contact information for the author/owner of a document or an article (i.e email address, URL, physical address, phone number).

- `<blockquote>` Tag: defines a section that is quoted from another source.
- `<bdo>` Tag: This is used to override the current text direction
- `<cite>` Tag: element usually renders in italic
- `<address>` Tag: usually renders in italic, and browsers will always add a line break before and after the `<address>` element.
- `<kbd>` Tag: it is used to define keyboard input. The content inside is displayed in the browser's default monospace font.
- `<samp>` Tag: it is used to define sample output from a computer program,
- `<var>` Tag : it is used to define a variable in programming or in a mathematical expression.
- A commonly used entity in HTML is the non-breaking space: ` `; A non-breaking space is a space that will not break into a new line.

Example:

```
<!DOCTYPE html>
<html>
<body>
<blockquote cite="https://en.wikipedia.org/wiki/Narendra_Modi">
Narendra Damodardas Modi;born 17 September 1950;is <cite>an Indian
politician</cite> serving as the 14th and current prime minister of India
since 2014.<bdo dir="rtl"> Modi</bdo> was the chief minister of Gujarat from
2001 to 2014 and is the Member of Parliament from Varanasi. He is a member of
the Bharatiya Janata Party (BJP) and of the<abbr title=" Rashtriya Swayamsevak
Sangh">RSS</abbr>, a right-wing Hindu nationalist paramilitary volunteer
organisation.
</blockquote>
<q>In every walk of life, you must have leaders. An education in the
spiritual world, in the labor field, in the agricultural field, we must have
leaders.</q>
<br>
<br>
<address>
Written by HeadMaster.<br>
Dr.NagaRaju<br>
HOD IT Department<br>
Bhimavaram<br>
</address>
</body>
</html>
```

Output:

Narendra Damodardas Modi; born 17 September 1950; is an Indian politician serving as the 14th and current prime minister of India since 2014. He was the chief minister of Gujarat from 2001 to 2014 and is the Member of Parliament from Varanasi. He is a member of the Bharatiya Janata Party (BJP) and of the RSS, a right-wing Hindu nationalist paramilitary volunteer organisation.

"In every walk of life, you must have leaders. An education in the spiritual world, in the labor field, in the agricultural field, we must have leaders."

*Written by HeadMaster:
Dr. NagaRaju
HOD IT Department
Bhimavaram*

Anchor tag: HTML Links are produced by using anchor tag. The Destination of the link is specified in the **href** attribute. The target attribute specifies where to open the linked document

- **_self** - Default. Opens the document in the same window/tab as it was clicked
- **_blank** - Opens the document in a new window or tab
- **_parent** - Opens the document in the parent frame
- **_top** - Opens the document in the full body of the window

Example:

```
<!DOCTYPE html>
<html>
<body>
<a href="https://www.movierulss.com/" target="_blank" >Pushpa Movie Link</a>
<!-- Anchor tag -->
</body>
</html>
```

Output:

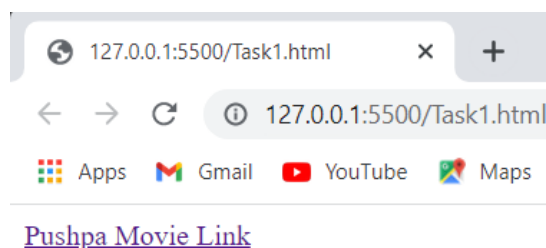
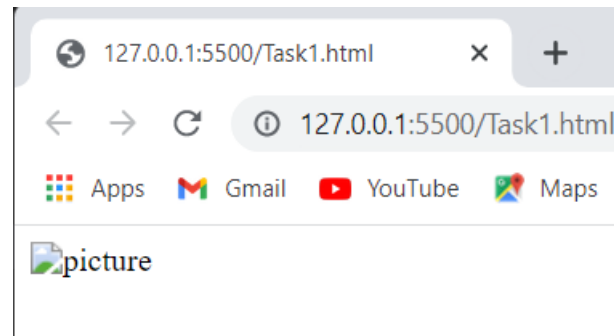


Image tag: HTML images are provided with the **** tag. The source file (**src**) is URL of the image or the path where image is located, alternative text (**alt**) specifies an alternate text for an image, **width**, and **height** are attributes for image tag specifies the width and height of the image. Background image from repeating itself, set the background-repeat property to no-repeat your images in a sub-folder, you must include the folder name in the src attribute. HTML allows animated gifs also. The **<picture>** tag gives web developers more flexibility in specifying image resources.

Example:

```
<!DOCTYPE html>
<html>
<body>
<img src="" alt="picture" width="104" height="142">
</body>
</html>
```

Output:



ImageMaps: The HTML <map> tag defines an image map. An image map is an image with clickable areas. The areas are defined with one or more <area> tags.

You must define the shape of the clickable area, and you can choose one of these values:

- rect - defines a rectangular region
- circle - defines a circular region
- poly - defines a polygonal region
- default - defines the entire region

ExampleCode:

```

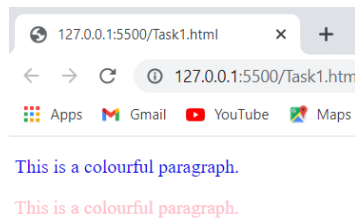
<map name="map">
<area shape="rect" coords="34,44,270,350" alt="1" href=" ">
<area shape="poly" coords="290,172,333,250" alt="2" href=" ">
<area shape="circle" coords="337,300,44" alt="3" href=" ">
</map>
```

Styles: The **style** attribute is used to add styles to an element, such as background- (image,size,color,cover,attachment),for text (color),border, font-family, size, and many more. Many colours are available.

Example1:

```
<!DOCTYPE html>
<html>
<body>
<p style="color:#0d10e9;font-size:auto;">This is a colourful paragraph.</p>
<p style="color:pink;">This is a colourful paragraph.</p>
</body>
</html>
```

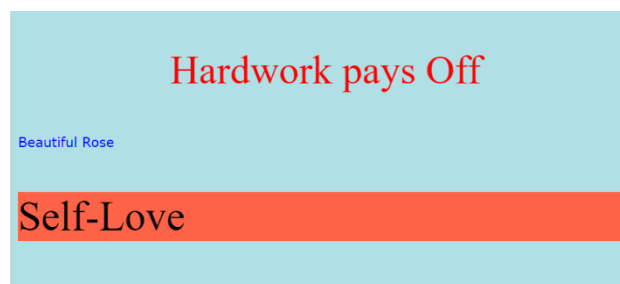
Output:



Example 2:

```
<!DOCTYPE html>
<html>
<body>
<body style="background-color:powderblue;">
<p style="color:red;font-size:300%;text-align:center;">Hardwork pays Off</p>
<p style="font-family:verdana;color:blue;">Beautiful Rose</p>
<p style="font-size:50px;background-color:tomato;">Self-Love</p>
</body>
</html>
```

Output:



RGB Color Values:

Each parameter (red, green, and blue) defines the intensity of the color with a value between 0 and 255.

This means that there are $256 \times 256 \times 256 = 16777216$ possible colors!

In HTML, a color can be specified as an RGB value, using this element:

rgb(red, green, blue)

HSL Color Values:

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

Saturation is a percentage value, 0% means a shade of gray, and 100% is the full color. Lightness is also a percentage value, 0% is black, and 100% is white.

hsl(hue, saturation, lightness)

Pre Tag: `<pre>` - preformatted text. The text inside a `<pre>` element is displayed in a fixed-width font and it will display both spaces and line breaks.

Example:

```
<!DOCTYPE html>
<html>
<body>
<pre>
I know that the day will come
when my sight of this earth shall be lost,
and life will take its leave in silence,
drawing the last curtain over my eyes.
Yet stars will watch at night,
and morning rise as before,
and hours heave like sea waves casting up pleasures and pains.
</pre>
</body>
</html>
```

Output:

```
I know that the day will come
when my sight of this earth shall be lost,
and life will take its leave in silence,
drawing the last curtain over my eyes.
Yet stars will watch at night,
and morning rise as before,
and hours heave like sea waves casting up pleasures and pains.
```

HTML Tables: Every table consists of rows and columns. This Content will always be placed inside `<table></table>` tag. `<td>` tag stands for table data and contains table data. `<tr>` stands for table row and starts with a `<tr>` and end with a `</tr>` tag. `<th>` stands for table heading in which elements are bold and centered, but you can change that with CSS.

- `<thead>`-Groups the header content in a table
- `<tbody>`-Groups the body content in a table
- `<tfoot>`-Groups the footer content in a table
- `<caption>`:we can add a caption that serves as a heading for the entire table.it should be inserted immediately after the `<table>` tag
- `text-align`: We can apply alignments of the data inside the table.
- `Colspan`: This attribute represents the number of columns to span means grouping into one column.
- `Rowspan`: This is same as `colspan` which is used to rows.

Styles in table:

- `border`: to add a border to a table.
- `border-color`: it sets the colour of the border.

- border-collapse: it will make the borders collapse into a single border.
- border-radius: the borders get rounded corners.
- border-style: This property can set the appearance of the border like **dotted,dashed,solid,double,groove,ridge,inset,outset,none,hidden**.
- Width, height: These properties specify the size of a table, row or column. We Use percentage as the size unit for a width which means how wide will this element be compared to its parent element, which in this case is the <body> element.
- border-spacing:
- padding: the space between cell borders and the content within a cell which are in pixels.
- The **border-bottom** property applied to all <tr> elements to get horizontal dividers.

Example Code:

```
<!DOCTYPE html>
<html>
  <body>
<table border="1" colspan="2" style=" border-radius: 5px;border-style:
dashed;border-collapse: collapse; padding:4px;color: palevioletred;">
  <caption>Monthly savings</caption>
  <tr>
    <th>Name</th>
    <th>ContactNumber</th>
    <th>Country</th>
  </tr>
  <tr>
    <td>Anushka Sharma</td>
    <td>2473265134</td>
    <td>India</td>
  </tr>
  <tr>
    <td>Bajirao</td>
    <td>2344345436</td>
    <td>India</td>
  </tr>
</table>
</body>
</html>
```

Output:

Monthly savings		
Name	ContactNumber	Country
Anushka Sharma	2473265134	India
Bajirao	2344345436	India

HTML Lists:

There are two types of lists.

1)Unordered HTML List: An unordered list starts with the tag. Each list item starts with the tag. The list items will be marked with bullets (small black circles) by default.

In Styles,The list-style-type property is used to define the style of the list item marker. It can have one of the following values:

- disc Sets the list item marker to a bullet (default)
- circle Sets the list item marker to a circle
- square Sets the list item marker to a square
- none The list items will not be marked

2)Ordered HTML List: An ordered list starts with the tag. Each list item starts with the tag. The list items will be marked with numbers by default and it contains <type> attribute which specifies {'1','a','A','i','I'}. By default, an ordered list will start from 1. If you want to start from a specified number, you can use <start> attribute.

.A description list is a list of terms, with a description of each term. The <dl> tag defines the description list, the <dt> tag defines the term (name), and the <dd> tag describes each term

Point to Remember: Lists can be nested

Example code:

```
<!DOCTYPE html>
<html>
<body>
<ul style="list-style-type:disc;">
  <li>Monday</li>
  <li>Tuesday</li>
  <ul>
    <li>MorningSession</li>
    <li>EveningSession</li>
  </ul>
  <li>Wednesday</li>
</ul>
<ol type='1'start="4">
  <li>Thursday</li>
  <li>Friday</li>
```

```

    <li>Saturday</li>
</ol>
<dl>
  <dt>DFC</dt>
  <dd>- Deccan Fried Chicken</dd>
  <dt>WHO</dt>
  <dd>- World Health Organisation</dd>
</dl>
</body>
</html>

```

Output:



HTML Block & In line Elements:

- A block-level element always starts on a new line and takes up the full width available.
- An inline element does not start on a new line and it only takes up as much width as necessary.
- The <div> element is a block-level and is often used as a container for other HTML elements
- The element is an inline container used to mark up a part of a text, or a part of a document
- We use id or class attribute to group and style specific elements which will be discussed more at CSS.

ExampleCode:

```

<!DOCTYPE html>
<html>
<body>
<div style="background-color:black;color:white;padding:20px;">
<h1>< div> Element Example</h1>
</div>

```

```
<p>In Vampire diaries,Damon has Attractive <span style="color:blue;font-weight:bold">blue</span> eyes and Stefan has Beautiful <span style="color:darkbrown;font-weight:bold">dark Brown</span> eyes.</p>

</body>
</html>
```

Output:

< div> Element Example

n Vampire diaries,Damon has Attractive **blue** eyes and Stefan has Beautiful **dark Brown** eyes.

HTML Iframe:

The HTML `<iframe>` tag specifies an inline frame.An inline frame is used to embed another document within the current HTML document.the height and width attributes to specify the size of the iframe . By using this Iframe, we can add youtube videos to webpage .

Syntax: `<iframe src="" title="description"></iframe>`

HTML Forms:

An HTML form is used to collect user input. The `<form>` element is used to create an HTML form for user input.it contains different elements like: text fields, checkboxes, radio buttons, submit buttons, etc.

`<label>` tag: it defines a label for form elements.

`<input>`Tag: An `<input>` element can be displayed in many ways, depending on the type attribute.

- `<input type="text">` = Displays a single-line text input field
- `<input type="radio">` = Displays a radio button (for selecting one of many choices)
- `<input type="checkbox">` = Displays a checkbox (for selecting zero or more of many choices)
- `<input type="submit">` = Displays a submit button (for submitting the form)
- `<input type="button">` = Displays a clickable button. Also `<button>` element defines a clickable button

The `<select>` element defines a drop-down list. The `<option>` elements defines an option that can be selected. The first item in the drop-down list is selected by default. the size attribute to specify the number of visible values. the multiple attribute to allow the user to select more than one value. datalist is a list of pre-defined options for an `<input>` element.

POINTS TO KNOW:

1. The `<fieldset>` element is used to group related data in a form.
2. The `<legend>` element defines a caption for the `<fieldset>` element.
3. The input readonly attribute specifies that an input field is read-only.
4. The input pattern attribute specifies a regular expression that the input field's value is checked against, when the form is submitted.

5. The input required attribute specifies that an input field must be filled out before submitting the form.
6. The input autocomplete attribute specifies whether a form or an input field should have autocomplete on or off. Autocomplete allows the browser to predict the value. When a user starts to type in a field, the browser should display options to fill in the field, based on earlier typed values.

Example Code:

REGISTRATION FORM

```
<!DOCTYPE html>
<html>
<body>
<form autocomplete="on">
  <label for="">Name:</label><br>
  <input type="text" id="" name="" placeholder="Enter Your Name"required><br>
  <label for="">Email Address:</label><br>
  <input type="email" pattern="[A-Z][a-z][0-9]{10}@" id="" name=""
placeholder="Enter Email"required>
  <br>
  <label for="">Gender:</label><br>
  <input type="radio" id="" name="" value="">
  <label for="">Male</label><br>
  <input type="radio" id="" name="" value="">
  <label for="">Female</label><br>
  <label for="">Languages Known:</label><br>
  <input type="checkbox" id="" name="" value="Bike">
  <label for="">English</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="">Telugu</label><br>
  <label for="cars">Choose your Interest:</label>
  <select id="" name="">
    <option value="Web-Development">Web-Development</option>
    <option value="Full-Stack Development">Full-Stack Development</option>
  </select>
  <input type="submit" onclick="alert('Registered Successfully!')"
value="Submit">
</form>
</body>
</html>
```

Output:

Name:

Email Address:

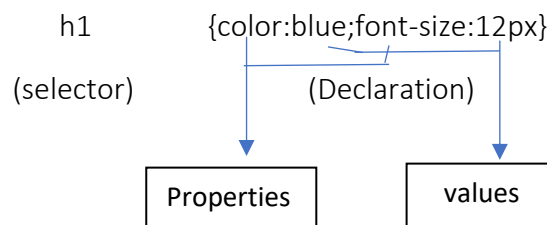
Gender:
☐ Male
☐ Female

Languages Known:
☐ English
☐ Telugu

Choose your Interest:

*****CSS*****

- CSS stands for Cascading Style Sheets.
- It is mainly used to add different Styles to html Webpages.
- It can control the layout of multiple web pages all at once.
- External stylesheets are stored in CSS files
- CSS is used to design styles for desired web pages, including the design, layout and variations in display for different devices and screen sizes.
- These are saved in external .css files.
- There are three ways of inserting a style sheet: External CSS(External styles are defined within the <link> element, inside the <head> section of an HTML page), Internal CSS(Internal styles are defined within the <style> element, inside the <head> section of an HTML page), Inline CSS(adding the style attribute to the relevant element).
- A CSS rule consists of a selector and a declaration block. The selector is the HTML element you want to style.
- The declaration block contains one or more declarations separated by semicolons. Each declaration includes a CSS property name and a value, separated by a colon.
- Declaration blocks are surrounded by curly braces.



Simple Code:

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  color: red;
  text-align: center;
}
</style>
</head>
<body>
<p>Welcome to CSS!</p>
</body>
</html>
```

Output:

Welcome to CSS!

CSS Selectors: CSS selector selects the HTML elements that we want to style. These are classified into five categories:

- Simple selectors (select elements based on name, id, class)
- Combinator selectors (select elements based on a specific relationship between them)
- Pseudo-class selectors (select elements based on a certain state)
- Pseudo-elements selectors (select and style a part of an element)
- Attribute selectors (select elements based on an attribute or attribute value)

The element selector selects HTML elements based on the element name. The id selector uses the id attribute of an HTML element to select a specific element. The id selector is used to select one unique element. To select an element with a specific id, write a hash (#), followed by the id of the element. An id name cannot start with a number.

```
<!DOCTYPE html>
<html>
<head>
<style>
#par1 {
  text-align: center;
  color: red;
}
</style>
</head>
<body>
<p id="par1">Welcome to CSS!</p>
<p>Best Example</p>
</body>
</html>
```

OUTPUT:

Welcome to CSS!

Best Example

- The class selector selects HTML elements with a specific class attribute, To select a specific class, write a dot (.), followed by the class name.
- The universal selector (*) selects all HTML elements on the page.
- Grouping Selectors: It will be better to group the similar elements through selectors, to minimize the code. To group selectors, separate each selector with a comma.
- CSS Comments are used to explain the code, and may help when you edit the source code at a later date. Comments are ignored by browsers.
- A CSS comment is placed inside the <style> element, and starts with /* and ends with */

Example Code:

```
<!DOCTYPE html>
<html>
<head>
<style>
h1 {
  background-color: green;
  text-align:center;
}
/*Element Selector*/
div {
  background-color: lightblue;
}
/*Universal Selector*/
*{
  text-size:20px;
}
/*Grouping Selectors*/
p,h2,h3{
  background-color: yellow;
}
</style>
</head>
<body>
<h1>CSS ExampleCode!</h1>
<div>Women's empowerment can be defined to<p> promoting women's sense of self-
worth, their ability to determine their own choices</p>, and their right to
influence social change for themselves and others.
</div>
</body>
</html>
```

Output:

CSS ExampleCode!

Women's empowerment can be defined to

promoting women's sense of self-worth, their ability to determine their own choices

, and their right to influence social change for themselves and others.

- The opacity property specifies the opacity/transparency of an element. It can take a value from 0.0 - 1.0. The lower value, the more transparent.
- 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. (background-repeat: repeat-x;) Repeats only horizontally,

(**background-repeat: repeat-y;**) Repeats only Vertically. The background image shows only once is also specified by the background-repeat property(**background-repeat: no-repeat;**). The background-position property is used to specify the position of the background image(**background-position: right top;**)

- The background-attachment property specifies whether the background image should scroll or be fixed (will not scroll with the rest of the page).

EXAMPLECODE:

```
body {
background-image: url("paper.gif");
background-repeat: no-repeat;
background-attachment: fixed;
opacity: 0.3;
}
```

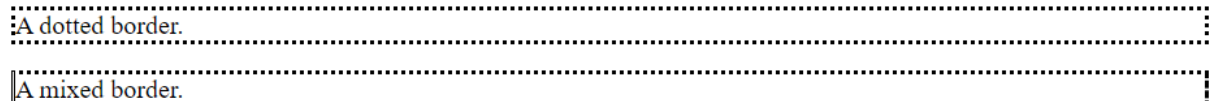
CSS Border Styles:

- The border-style property specifies what kind of border to display.
- The border styles same as HTML styles like Colour,size,width,radius etc

EXAMPLE: <!DOCTYPE html>

```
<html>
<head>
<style>
p.dotted {border-style: dotted;}
p.mix {border-style: dotted dashed solid double;}
</style>
</head>
<body>
<p class="dotted">A dotted border.</p>
<p class="mix">A mixed border.</p>
</body>
</html>
```

- Output:



CSS Margins:

The CSS margin properties are used to create space around elements, outside of any defined borders. There are three margin values:

margin: 25px 50px 75px;

top margin is 25px, right and left margins are 50px, bottom margin is 75px

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).

All the padding properties can have the following values:

1. length - specifies a padding in px, pt, cm, etc.
2. % - specifies a padding in % of the width of the containing element
3. inherit - specifies that the padding should be inherited from the parent element

If the padding property has two values:

padding: 25px 50px;

1. top and bottom paddings are 25px
2. right and left paddings are 50px

ExampleCode:

```
<!DOCTYPE html>
<html>
<head>
<style>
p {
  border: 2px solid black;
  outline: #4CAF50 solid 10px;
  margin: auto;
  padding: 20px;
  text-align: center;
}
</style>
</head>
<body>
<h2>CSS:</h2>
<p>Honesty is the best policy</p>
</body>
</html>
```

Output:



CSS Outline Style:

The outline-style property specifies the style of the outline, and can have one of the following values:

1. dotted - Defines a dotted outline(.....)
2. dashed - Defines a dashed outline(-----)
3. solid - Defines a solid outline
4. double - Defines a double outline
5. groove - Defines a 3D grooved outline
6. ridge - Defines a 3D ridged outline
7. inset - Defines a 3D inset outline
8. outset - Defines a 3D outset outline
9. none - Defines no outline
10. hidden - Defines a hidden outline.
11. Example: `p.dotted {outline-style: dotted;}`
12. The outline-color property is used to set the color of the outline.
13. Example: `p.ex1 {
border: 8px solid black;
outline-style: solid; outline-color: pink; outline-offset: 15px;
}`
14. The outline-offset property adds space between an outline and the edge/border of an element. The space between an element and its outline is transparent.

CSS TEXT PROPERTIES:

The text-transform property is used to specify uppercase and lowercase letters in a text. The text-indent property is used to specify the indentation of the first line of a text. The letter-spacing property is used to specify the space between the characters in a text. The line-height property is used to specify the space between lines. The word-spacing property is used to specify the space between the words in a text. The `text-shadow` property adds shadow to text. In its simplest use, you only specify the horizontal shadow (2px) and the vertical shadow (2px):

Text shadow effect!

Example: `p.capitalize {
text-transform: capitalize; text-indent: 50px;
}`

CSS Alignments: The text-align property is used to set the horizontal alignment of a text. The text-align-last property specifies how to align the last line of a text. The direction and unicode-bidi properties can be used to change the text direction of an element. Direction: Specifies the text direction/writing direction. unicode-bidi: Used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document. vertical-align: Sets the vertical alignment of an element.

CSS text-decoration Properties:

text-decoration-Sets all the text-decoration properties in one declaration

text-decoration-color : Specifies the color of the text-decoration

text-decoration-line : Specifies the kind of text decoration to be used (underline, overline, etc.)

text-decoration-style : Specifies the style of the text decoration (solid, dotted, etc.)

text-decoration-thickness : Specifies the thickness of the text decoration line.

CSS Fonts :

Serif - Times New Roman, Georgia, Garamond

Sans-serif – Arial, Verdana, Helvetica

Monospace - Courier New, Lucida Console, Monaco

Cursive - Brush Script MT, Lucida Handwriting

Fantasy – Copperplate, Papyrus

- The font-style property is mostly used to specify italic text.
- This property has two values:
 1. normal - The text is shown normally
 2. italic - The text is shown in italics

The font-weight property specifies the weight of a font i.e, normal or bold etc.

CSS Link Styles:

We can include our desired styles and Fonts not only for links but also used for Some HTML elements like Buttons, lists, tables, Paragraphs ..etc in these Properties. The four links states are:

- a:link - a normal, unvisited link
- a:visited - a link the user has visited
- a:hover - a link when the user mouses over it
- a:active - a link the moment it is clicked

ExampleCode:

```
<!DOCTYPE html>

<html>
<head>
<style>
/* unvisited link */
a:link {
  color: red;
}

/* visited link */
a:visited {
  color: green;
}

/* mouse over link */
a:hover {
  color: hotpink;
}

/* selected link */
a:active {
  color: blue;
}
</style>
</head>
<body>
<p><b><a href=" " target="_blank">This is a link</a></b></p>
</body>
</html>
```

- Output:



CSS CURSOR PROPERTIES: Mouse over the words to change the cursor.

ExampleCode:

```
<!DOCTYPE html>
<html>
<body>
<span style="cursor:auto">auto</span><br>
<span style="cursor:crosshair">crosshair</span><br>
<span style="cursor:default">default</span><br>
<span style="cursor:e-resize">e-resize</span><br>
<span style="cursor:help">help</span><br>
<span style="cursor:move">move</span><br>
<span style="cursor:n-resize">n-resize</span><br>
<span style="cursor:ne-resize">ne-resize</span><br>
<span style="cursor:nw-resize">nw-resize</span><br>
<span style="cursor:pointer">pointer</span><br>
<span style="cursor:progress">progress</span><br>
<span style="cursor:s-resize">s-resize</span><br>
<span style="cursor:se-resize">se-resize</span><br>
<span style="cursor:sw-resize">sw-resize</span><br>
<span style="cursor:text">text</span><br>
<span style="cursor:w-resize">w-resize</span><br>
<span style="cursor:wait">wait</span><br>
</body>
</html>
```

Copy and paste it and see the results to understand them clearly.

CSS DisplayProperty:

The display property is the most important CSS property for controlling layout. The display property specifies if/how an element is displayed. Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline.

Block-level Elements:

A block-level element always starts on a new line and takes up the full width available (stretches out to the left and right as far as it can).

CSS LAYOUT PROPERTIES: The position property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky). The position property specifies the type of positioning method used for an element. There are five different position values:

1. Static - HTML elements are positioned static by default. Static positioned elements are not affected by the top, bottom, left, and right properties. An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page
2. Relative - An element with position: relative; is positioned relative to its normal position. Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

3. Fixed - An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element. A fixed element does not leave a gap in the page where it would normally have been located.
4. Absolute - An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed). However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.
5. Sticky - An element with position: sticky; is positioned based on the user's scroll position. A sticky element toggles between relative and fixed, depending on the scroll position. It is positioned relative until a given offset position is met in the viewport - then it "sticks" in place (like position: fixed).

z-index Property: When elements are positioned, they can overlap other elements. The **z-index** property specifies the stack order of an element (which element should be placed in front of, or behind, the others).

CSS Overflow: The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified area. The overflow property has the following values:

1. visible - Default. The overflow is not clipped. The content renders outside the element's box
2. hidden - The overflow is clipped, and the rest of the content will be invisible
3. scroll - The overflow is clipped, and a scrollbar is added to see the rest of the content
4. auto - Similar to scroll, but it adds scrollbars only when necessary

Note: The overflow property only works for block elements with a specified height.

CSS Float Property:

The float property is used for positioning and formatting content. The float property can have one of the following values:

1. left - The element floats to the left of its container
2. right - The element floats to the right of its container
3. none - The element does not float (will be displayed just where it occurs in the text). This is default
4. inherit - The element inherits the float value of its parent.

Note: The clear property specifies what should happen with the element that is next to a floating element. The overflow: auto clearfix works well as long as you are able to keep control of your margins and padding (else you might see scrollbars).

Example: `.clearfix { overflow: auto; }`

CSS Combinators:

A combinator is something that explains the relationship between the selectors. There are four different combinators in CSS:

- descendant selector (space) : it matches all elements that are descendants of a specified element.
- child selector (>) : it selects all elements that are the children of a specified element.
- adjacent sibling selector (+) : it used to select an element that is directly after another specific element. Sibling elements must have the same parent element, and "adjacent" means "immediately following".
- general sibling selector (~) : it selects all elements that are next siblings of a specified element.

CSS 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
- a:hover MUST come after a:link and a:visited in the CSS definition in order to be effective!
a:active MUST come after a:hover in the CSS definition in order to be effective! Pseudo-class names are not case-sensitive.

Selector	Example	Description
::after	p::after	Insert content after every <p> element
::before	p::before	Insert content before every <p> element
::first-letter	p::first-letter	Selects the first letter of every <p> element
::first-line	p::first-line	Selects the first line of every <p> element
::selection	p::selection	Selects the portion of an element that is selected by a user

CSS Images: See and use the example and Understand the Concept.

```
<!DOCTYPE html>  
<html>
```



```

<head>
<style>
div.gallery {
    margin: 5px;
    border: 1px solid #ccc;
    float: left;
    width: 180px;
}

div.gallery:hover {
    border: 1px solid #777;
}

div.gallery img {
    width: 100%;
    height: auto;
}

div.desc {
    padding: 15px;
    text-align: center;
}
</style>
</head>
<body>
<div class="gallery">
    <a target="_blank" href="img_5terre.jpg">
        
    </a>
    <div class="desc">Add a description of the image here</div>
</div>
<div class="gallery">
    <a target="_blank" href="img_forest.jpg">
        
    </a>
    <div class="desc">Add a description of the image here</div>
</div>
<div class="gallery">
    <a target="_blank" href="img_lights.jpg">
        
    </a>
    <div class="desc">Add a description of the image here</div>
</div>

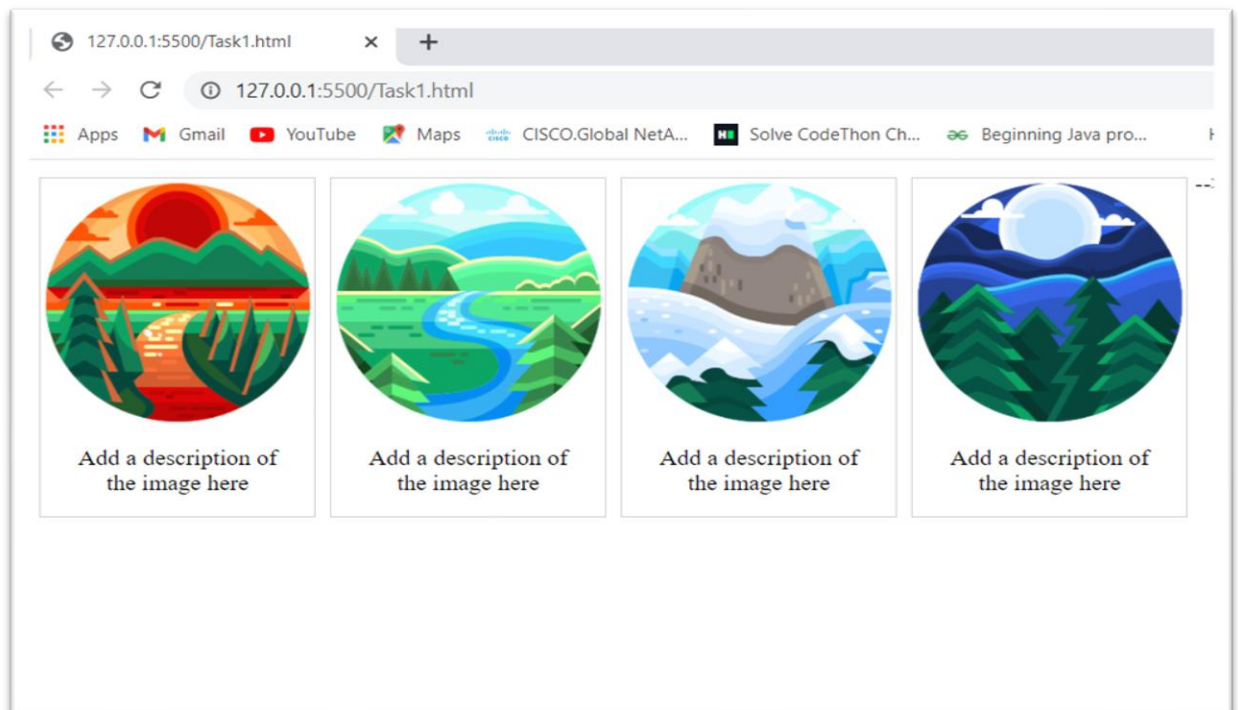
```

```

</div>
<div class="gallery">
  <a target="_blank" href="img_mountains.jpg">
    
  </a>
  <div class="desc">Add a description of the image here</div>
</div>
</body>
</html>

```

Output:



CSS counters:

these are like "variables". The variable values can be incremented by CSS rules (which will track how many times they are used).

To work with CSS counters we will use the following properties:

1. counter-reset - Creates or resets a counter
2. counter-increment - Increments a counter value
3. content - Inserts generated content
4. counter() or counters() function - Adds the value of a counter to an element.

To use a CSS counter, it must first be created with counter-reset.

Note: The CSS math functions allow mathematical expressions to be used as property values. Here, we will explain the calc(), max() and min() functions.

CSS [attribute] Selector: The [attribute] selector is used to select elements with a specified attribute.

Selector	Example	Example description
[attribute]	[target]	Selects all elements with a target attribute
[attribute=value]	[target=_blank]	Selects all elements with target="_blank"
[attribute~=value]	[title~=flower]	Selects all elements with a title attribute containing the word "flower"
[attribute =value]	[lang =en]	Selects all elements with a lang attribute value starting with "en"
[attribute^=value]	a[href^="https"]	Selects every <a> element whose href attribute value begins with "https"
[attribute\$=value]	a[href\$=".pdf"]	Selects every <a> element whose href attribute value ends with ".pdf"
[attribute*=value]	a[href*="w3schools"]	Selects every <a> element whose href attribute value contains the substring "w3schools"

CSS gradients:

It will let you display smooth transitions between two or more specified colors.

CSS defines three types of gradients:

- Linear Gradients (goes down/up/left/right/diagonally)
- Radial Gradients (defined by their center)
- Conic Gradients (rotated around a center point)

Syntax:

```
background-image: linear-gradient(direction, color-stop1, color-stop2, ...);
```

CSS Radial Gradients:

A radial gradient is defined by its center. To create a radial gradient you must also define at least two color stops.

The size parameter defines the size of the gradient. It can take four values:

- closest-side
- farthest-side
- closest-corner
- farthest-corner

Syntax:

background-image: radial-gradient(shape size at position, start-color, ..., last-color);

CSS Conic Gradients: A conic gradient is a gradient with color transitions rotated around a center point. To create a conic gradient you must define at least two colors.

Syntax:

background-image: conic-gradient([from angle] [at position,] color [degree], color [degree], ...);

Note:

1. Creating Pie Charts: Just add border-radius: 50% to make the conic gradient look like a pie.
2. Repeating a Conic Gradient: The repeating-conic-gradient() function is used to repeat conic gradients

CSS BOX-SHADOW :

1. The CSS **box-shadow** property is used to apply one or more shadows to an element. The **color** parameter defines the color of the shadow.
2. The **blur** parameter defines the blur radius. The higher the number, the more blurred the shadow will be.
3. The **spread** parameter defines the spread radius. A positive value increases the size of the shadow, a negative value decreases the size of the shadow.
4. The **inset** parameter changes the shadow from an outer shadow (outset) to an inner shadow.

Points to Know: The CSS text-overflow property specifies how overflowed content that is not displayed should be signaled to the user.

- The CSS word-wrap property allows long words to be able to be broken and wrap onto the next line.
- The CSS word-break property specifies line breaking rules.
- The CSS writing-mode property specifies whether lines of text are laid out horizontally or vertically.
- The box-reflect property is used to create an image reflection. The value of the box-reflect property can be: below, above, left , or right.
- The CSS mask-image property specifies a mask layer image. The mask layer image can be a PNG image, an SVG image, a CSS gradient, or an SVG <mask> element.
- The CSS filter property adds visual effects (like blur and saturation) to an element.
- CSS 2D Transforms: CSS transforms allow you to move, rotate, scale, and skew elements. With the CSS **transform** property you can use the following 2D transformation methods:
 1. **translate()** : it moves an element from its current position (according to the parameters given for the X-axis and the Y-axis).
 2. **rotate()** : it rotates an element clockwise or counter-clockwise according to a given degree.
 3. **scale()**: it increases or decreases the size of an element (according to the parameters given for the width and height).
 4. **scaleX()** : for Accessing width.
 5. **scaleY()** : for Accessing height.
 6. **skewX()** : skews an element along the X-axis by the given angle.
 7. **skewY()** : skews an element along the Y-axis by the given angle.
 8. **skew()** : skews an element along the X and Y-axis by the given angles.
 9. **matrix()**: it is a method combines all the 2D transform methods into one.

CSS 3D Transforms Methods:

With the CSS transform property you can use the following 3D transformation methods:

1. rotateX() – this method rotates an element around its X-axis at a given degree.
2. rotateY() – this method rotates an element around its Y-axis at a given degree
3. rotateZ()- this method rotates an element around its Z-axis at a given degree

CSS transitions:

These allows you to change property values smoothly, over a given duration

1. transition
2. transition-delay: property specifies a delay (in seconds) for the transition effect.
3. transition-duration: Specifies how many seconds or milliseconds a transition effect takes to complete
4. transition-property: Specifies the name of the CSS property the transition effect is for.

5. transition-timing-function. Specify the Speed Curve of the Transition. The transition-timing-function property specifies the speed curve of the transition effect. The transition-timing-function property can have the following values:
- ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
 - linear - specifies a transition effect with the same speed from start to end
 - ease-in - specifies a transition effect with a slow start
 - ease-out - specifies a transition effect with a slow end
 - ease-in-out - specifies a transition effect with a slow start and end
 - cubic-bezier(n,n,n,n) - lets you define your own values in a cubic-bezier function

CSS ToolTips:

A tooltip is often used to specify extra information about something when the user moves the mouse pointer over an element. Explore it to learn more about it.

Simple Pagination: If you have a website with lots of pages, you may wish to add some sort of pagination to each page.

EXAMPLE CODE:

```
<!DOCTYPE html>
<html>
<head>
<style>
.pagination {
  display: inline-block;
}
.pagination a {
  color: black;
  float: left;
  padding: 8px 16px;
  text-decoration: none;
}
.pagination a.active {
  background-color: #4CAF50;
  color: white;
}
.pagination a:hover:not(.active) {background-color: #ddd;}
</style>
</head>
```

```

<body>
<h2>Active and Hoverable Pagination</h2>
<p>Move the mouse over the numbers.</p>
<div class="pagination">
  <a href="#">&laquo;</a>
  <a href="#">1</a>
  <a class="active" href="#">2</a>
  <a href="#">3</a>
  <a href="#">4</a>
  <a href="#">5</a>
  <a href="#">6</a>
  <a href="#">&raquo;</a>
</div>
</body>
</html>

```

Output:

Active and Hoverable Pagination

Move the mouse over the numbers.

« 1 2 3 4 5 6 »

Note:

- The `resize` property specifies if (and how) an element should be resizable by the user. Example: `div { resize: horizontal; overflow: auto; }`
- The `outline-offset` property adds space between an outline and the edge or border of an element.

CSS Flexbox Layout Module:

Before the Flexbox Layout module, there were four layout modes:

- Block, for sections in a webpage
- Inline, for text
- Table, for two-dimensional table data
- Positioned, for explicit position of an element

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

Examplecode:

```

<!DOCTYPE html>

```

```

<html>
<head>
<style>
.flex-container {
  display: flex;
  background-color: grey;
}

.flex-container > div {
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
</style>
</head>
<body>

<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>

</body>
</html>

```

OUTPUT:



Some Other Flex Properties:

align-content: Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines

align-items: Vertically aligns the flex items when the items do not use all available space on the cross-axis

display: Specifies the type of box used for an HTML element

flex-direction: Specifies the direction of the flexible items inside a flex container

flex-flow: A shorthand property for flex-direction and flex-wrap

flex-wrap: Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line

justify-content : Horizontally aligns the flex items when the items do not use all available space on the main-axis

****Bootstrap****

Bootstrap 5 is the newest version of Bootstrap, which is the most popular HTML, CSS, and JavaScript framework with new components, faster stylesheet and more responsiveness.

Steps to create webpage using Bootstrap:

1. Bootstrap 5 uses HTML elements and CSS properties that require the HTML5 doctype.
2. Bootstrap 5 is mobile-first: Bootstrap 5 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework. To ensure proper rendering and touch zooming, add the following <meta> tag inside the <head> element.
3. Containers:
Bootstrap 5 also requires a containing element to wrap site contents.

Basic code: Copy & Paste the code and understand it clearly.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min
.js"></script>
</head>
<body>  <-- Upto This code is common for all bootstrap codes-->

<div class="container">
  <h1>Bootstrap</h1>
  <p>inside .container class</p>
  <p>Resize the browser window to see that the container width will change at
different breakpoints.</p>
</div>
</body>
</html>
```

Output:

Bootstrap

inside .container class

Resize the browser window to see that the container width will change at different breakpoints.

Containers are used to pad the content inside of them, and there are two container classes available:

- The .container class provides a responsive fixed width container
- SyntaxExample: `<div class="container">`
`<h1>Example</h1>`
`<p>For Container class</p>`
`</div>`
- The .container-fluid class provides to create a full width container, that will always span the entire width of the screen (width is always 100%)
- SyntaxExample: `<div class="container-fluid">`
`<h1> Example For </h1>`
`<p> Container-Fluid class </p>`
`</div>`

Container Padding:

By default, containers have left and right padding, with no top or bottom padding. Therefore, we often use spacing utilities, such as extra padding and margins to make them look even better.

```
ExampleCode: <!DOCTYPE html>

<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
  <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"></script>
</head>
<body>

<div class="container pt-5">
  <h1>An Extra Top padding</h1>
</div>
```

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

Output:

An Extra Top padding

Borders and Colours:

You can also use the `.container-sm|md|lg|xl` classes to determine when the container should be responsive.

For example: create a row (`<div class="row">`). Then, add the desired number of columns (tags with appropriate `.col-*-*` classes). The first star (*) represents the responsiveness: sm, md, lg, xl or xxl, while the second star represents a number, which should add up to 12 for each row.

Examplecode: `<div class="container-sm">Small sized text</div>`

Output: small sized text

The Bootstrap 5 grid system has six classes:

- `.col-` (extra small devices - screen width less than 576px)
- `.col-sm-` (small devices - screen width equal to or greater than 576px)
- `.col-md-` (medium devices - screen width equal to or greater than 768px)
- `.col-lg-` (large devices - screen width equal to or greater than 992px)
- `.col-xl-` (xlarge devices - screen width equal to or greater than 1200px)
- `.col-xxl-` (xxlarge devices - screen width equal to or greater than 1400px)

For example: instead of adding a number to each col, let bootstrap handle the layout, to create equal width columns: two "col" elements = 50% width to each col, while three cols = 33.33% width to each col. Four cols = 25% width, etc. You can also use `.col-sm|md|lg|xl|xxl` to make the columns responsive.

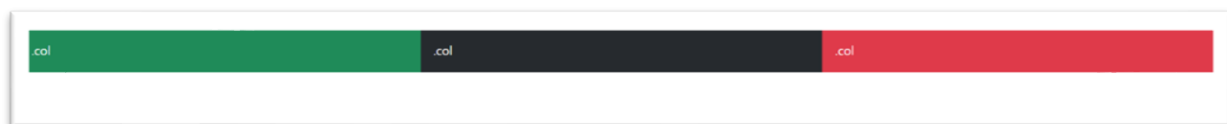
```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
```

```

<meta name="viewport" content="width=device-width, initial-scale=1">
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min
.js"></script>
</head>
<body>
<div class="container-fluid mt-3">
<div class="row">
<div class="col p-3 bg-success text-white">.col</div>
<div class="col p-3 bg-dark text-white">.col</div>
<div class="col p-3 bg-danger text-white">.col</div>
</div>
</div>
</div>
</body>

```

Output:



Headings:

We use .h1 to .h6 classes on other elements to make them behave as headings. Display headings are used to stand out more than normal headings (larger font-size and lighter font-weight), and there are six classes to choose from: .display-1 to .display-6.

Using Syntax:

```

<div class="container mt-3">
<p class="h1">h1 heading</p>
<p class="h2">h2 heading</p>
<h1 class="display-1">Display 1</h1>
<h1 class="display-2">Display 2</h1>
<h1 class="display-3">Display 3</h1>
<h1 class="display-4">Display 4</h1>
<h1 class="display-5">Display 5</h1>
<h1 class="display-6">Display 6</h1>
</div>

```

h1 heading

h2 heading

Display 1

Display 2

Display 3

Display 4

Display 5

Display 6

<Mark> tag:

Bootstrap 5 will style <mark> and .mark with a yellow background color and some padding.

Using Syntax:

```
<div class="container mt-3">
  <p> <mark>highlight</mark></p>
</div>
```

Output:

Highlight

Bootstrap Colours & Background colours:

- The classes for text colors are: `.text-muted`, `.text-primary`, `.text-success`, `.text-info`, `.text-warning`, `.text-danger`, `.text-secondary`, `.text-white`, `.text-dark`, `.text-body` (default body color /often black) and `.text-light`.
- You can also add 50% opacity for black or white text with the `.text-black-50` or `.text-white-50` classes.
- The classes for background colors are: `.bg-primary`, `.bg-success`, `.bg-info`, `.bg-warning`, `.bg-danger`, `.bg-secondary`, `.bg-dark` and `.bg-light`.
- Syntax Example: Refer Example at Container-Fluid .

Bootstrap Tables:

A basic Bootstrap 5 table has a light padding and horizontal dividers. The `.table` class adds basic styling to a table. The `.table-striped` class adds zebra-stripes to a table. The `.table-responsive` class adds a scrollbar to the table when needed (when it is too big horizontally). The `.table-bordered` class adds borders on all sides of the table and cells. The `.table-hover` class adds a hover effect (grey background color) on table rows. The `.table-*color` class adds a black background to the table. The `.table-responsive` class adds a scrollbar to the table when needed (when it is too big horizontally)

1. `.table-primary` : Blue: Indicates an important action
2. `.table-success` : Green: Indicates a successful or positive action
3. `.table-danger` : Red: Indicates a dangerous or potentially negative action
4. `.table-info` : Light blue: Indicates a neutral informative change or action
5. `.table-warning` : Orange: Indicates a warning that might need attention
6. `.table-active` : Grey: Applies the hover color to the table row or table cell
7. `.table-secondary` : Grey: Indicates a slightly less important action
8. `.table-light` : Light grey table or table row background
9. `.table-dark` : Dark grey table or table row background)The `.table-borderless` class removes borders from the table.

```
SyntaxExample: <!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```

<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet">
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min
.js"></script>
</head>
<body>
<div class="container mt-3">
  <table class="table table-striped table-grey table-hover">
    <thead>
      <tr>
        <th>Name</th>
        <th>BirthYear</th>
        <th>Email</th>
      </tr>
    </thead>
    <tbody>

```

Output:

Name	BirthYear	Email
July	1976	july@example.com
June	2003	june@example.com
May	2003	may@example.com

Bootstrap Images:

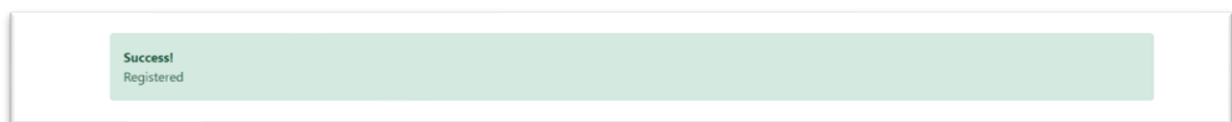
- The `.rounded` class adds rounded corners to an image.
- Syntax: ``
- The `.rounded-circle` class shapes the image to a circle:
- Syntax: ``
- The `.img-thumbnail` class shapes the image to a thumbnail (bordered)
- Syntax: ``
- **Alignment of Images:** Float an image to the left with the `.float-start` class or to the right with `.float-end`. Center an image by adding the utility classes `.mx-auto` (margin:auto) and `.d-block` (display:block) to the image.

- Images come in all sizes. **Responsive images** automatically adjust to fit the size of the screen. Create responsive images by adding an `.img-fluid` class to the `` tag. The image will then scale nicely to the parent element. The `.img-fluid` class applies `max-width: 100%;` and `height: auto;` to the image.

Bootstrap Alerts: Alerts are created with the `.alert` class, followed by one of the contextual classes `.alert-success`, `.alert-info`, `.alert-warning`, `.alert-danger`, `.alert-primary`, `.alert-secondary`, `.alert-light` or `.alert-dark`. Add the `(class= "alert-link")` to any links inside the alert box to create "matching colored links". To close the alert message, add `class="alert-dismissible"` to the alert container. Then add `class="btn-close"` and `data-bs-dismiss="alert"` to a link or a button element (when you click on this the alert box will disappear).

Example-1: `<div class="alert alert-success">Success!Registered</div>`

Output:



Bootstrap Buttons:

Use the `class="btn-lg"` for large buttons or `class="btn-sm"` for small buttons. To create a block level button that spans the entire width of the parent element, use the `.d-grid "helper"` class on the parent element. The class `.active` makes a button appear pressed, and the `disabled` attribute makes a button unclickable. Note that `<a>` elements do not support the `disabled` attribute and must therefore use the `.disabled` class to make it visually appear disabled. If you have many block-level buttons, you can control the space between them with the `class="d-grid gap-3"`. You can also add "spinners" to a button. Use the `spinner-grow` class if you want the spinner/loader to grow instead of "spin". Use the `.spinner-grow` class if you want the spinner/loader to grow instead of "spin".

For outline Buttons, Example-1: `<button type="button" class="btn btn-outline-success">Success</button>`

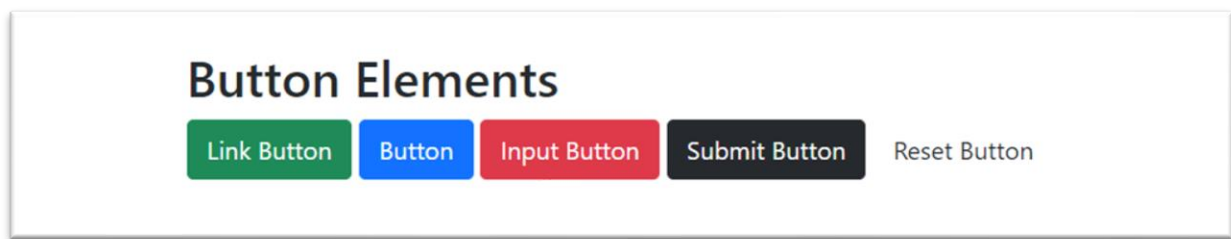
ExampleCode: `<div class="container mt-3">`

```

    <h2>Button Elements</h2>
    <a href="#" class="btn btn-success">Link Button</a>
    <button type="button" class="btn btn-primary">Button</button>
    <input type="button" class="btn btn-danger" value="Input Button">
    <input type="submit" class="btn btn-dark" value="Submit Button">
    <input type="reset" class="btn btn-light" value="Reset Button">
</div>

```

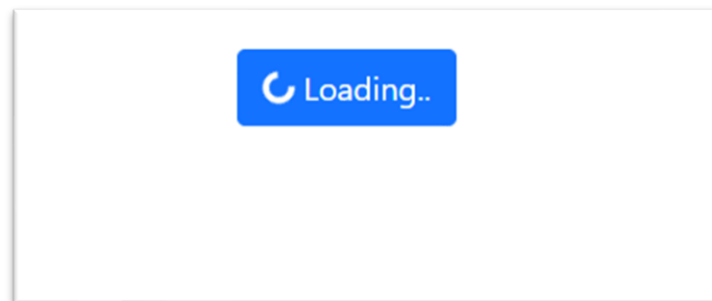
Output:



Example-2: `<button class="btn btn-primary">`

```
    <span class="spinner-border spinner-border-sm"></span>
    Loading..
</button>
```

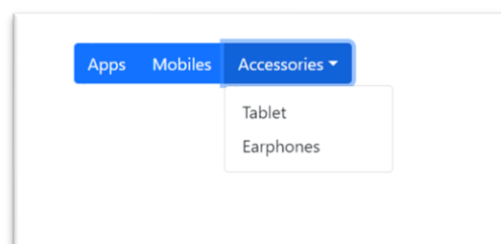
Output:



Example-3:

```
<div class="container mt-3">
  <div class="btn-group">
    <button type="button" class="btn btn-primary">Apps</button>
    <button type="button" class="btn btn-primary">Mobiles</button>
    <div class="btn-group">
      <button type="button" class="btn btn-primary dropdown-toggle"
data-bs-toggle="dropdown">Accessories</button>
      <ul class="dropdown-menu">
        <li><a class="dropdown-item" href="#">Tablet</a></li>
        <li><a class="dropdown-item" href="#">Earphones</a></li>
      </ul>
    </div>
  </div>
</div>
```

Output:



Bootstrap Badges:

Badges are used to add additional information to any content. Use `class="badge "` together with a contextual class (like `.bg-secondary`) within `` elements to create rectangular badges. The badges scale must match the size of the parent element. Use the `rounded-pill` class to make the badges more round.

ExampleCode:

```
<div class="container mt-3">
  <span class="badge bg-danger">Danger</span>
  <span class="badge rounded-pill bg-primary">Primary</span>
  <button type="button" class="btn btn-dark">
    Messages <span class="badge bg-danger">4</span>
  </button>
</div>
```

Output:



Bootstrap Progress Bar:

A progress bar can be used to show how far a user is in a process. To create a default progress bar, add a `progress` class to a container element and add the `progress-bar` class to its child element. Use the CSS `width` property to set the width of the progress bar. Use the CSS `height` property to change it. Use the `progress-bar-striped` class to add stripes to the progress bars. Add the `progress-bar-animated` class to animate the progress bar.

Examplecode:

```
<div class="container mt-3">
  <div class="progress" style="height:10px">
    <div class="progress-bar" style="width:40%;height:10px">40%</div>
  </div>
  <br>
  <div class="progress">
    <div class="progress-bar progress-bar-striped bg-success"
style="width:50%">50%</div>
  </div>
  <br>
  <div class="progress" style="height:30px">
    <div class="progress-bar progress-bar-striped progress-bar-animated
bg-dark" style="width:60%;height:30px">
    <div class="progress-bar progress-bar-striped progress-bar-animated
bg-danger" style="width:20%">
```

```

    Partially completed
  </div></div>
</div>
</div>

```

Output:



Bootstrap Pagination:

create a pagination, add the .pagination class to an element. Then add the page-item to each element and a .page-link class to each link inside .The active class is used to "highlight" the current page. The disabled class is used for un-clickable links.

Bootstrap Borders:

Use the **border** classes to add or remove borders from an element. Use **border-1** to **border-5** to change the width of the border. Add a color to the border with any of the contextual border color classes. Add rounded corners to an element with the **rounded** classes. Float an element to the right with the **float-end** class or to the left with **float-start**, and clear floats with the **clearfix** class.Know more, by Trying them with examples.

Bootstrap Checkbox:

To style checkboxes, use a wrapper element with **class="form-check"** to ensure proper margins for labels (class="form-check-label") and checkboxes. Then, add the .form-check-label class to label elements, and form-check-input to style checkboxes properly inside the .form-check container. Use the checked attribute if you want the checkbox to be checked by default. For toggleSwitches, use **class="form-check form-switch"**

Bootstrap Ranges:

Add the .form-range class to the input element with type="range", By default, the interval between the range numbers is 1. You can change it by using the step attribute. By default, the minimum value is 0 and maximum value is 100. You can use the min and/or max attribute change it.

Bootstrap Input-groups:

The input-group class is a container to enhance an input by adding an icon, text or a button in front or behind the input field as a "help text". To style the specified help text, use the .input-group-text class.

Bootstrap Lists:

```

<div class="container mt-3">
  <ul class="list-group list-group-vertical">
    <li class="list-group-item active">Active item</li>

```

```

        <a href="#" class="list-group-item list-group-item-danger list-
group-item-action">Junks <span class="badge bg-primary rounded-
pill">99</span></a>
        <a href="#" class="list-group-item disabled">Disabled item</a>
    </ul>
</div>

```

Output:



Bootstrap Cards:

A basic card is created with the `card` class, and content inside the card has a `card-body` class. The `card-header` class adds a heading to the card and the `card-footer` class adds a footer to the card. Add `card-img-top` or `card-img-bottom` to an `` to place the image at the top or at the bottom inside the card. Note that we have added the image outside of the `card-body` to span the entire width. Turn an image into a card background and use `.card-img-overlay` to add text on top of the image.

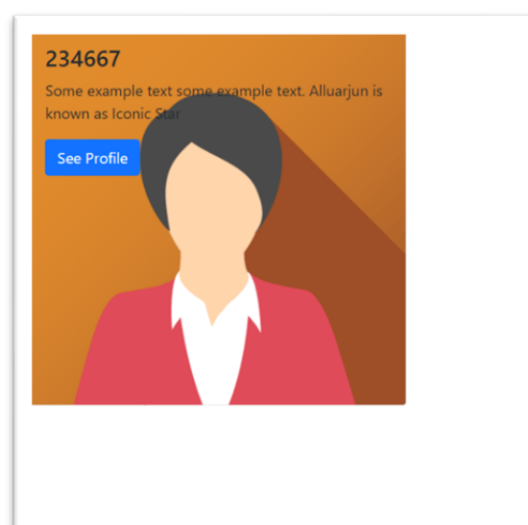
ExampleCode:

```

<div class="card" style="width:400px">
    <div class="card-body-sm">
        <div class="card-img-overlay"><h4 class="card-title">234667</h4>
        <p class="card-text">Some example text some example text. Alluarjun
is known as Iconic Star</p>
        <a href="#" class="btn btn-primary">See Profile</a>
    </div>
    
    </div></div>
</div>

```

Output:



Bootstrap Navigation Properties:

For a Horizontal menu, add the `nav` class to a `` element, followed by `nav-item` for each `` and add the `nav-link` class to their links or for buttons also. Convert the nav menu into navigation pills with the **`nav-pills`** class. To make the tabs toggleable, add the **`data-toggle="tab"`** attribute to each link. Then add a tab-pane class with a unique ID for every tab and wrap them inside a `<div>` element with **`class tab-content`**. If you want the tabs to fade in and out when clicking on them, add the `fade` class to tab-pane. Know more, by Trying them with examples.

Bootstrap Offcanvas:

Offcanvas is similar to modals (hidden by default and shown when activated), except that is often used as a sidebar navigation menu. The `.offcanvas` class creates the offcanvas sidebar.

The `.offcanvas-start` class positions the offcanvas, and makes it 400px wide. See examples below for more positioning classes. The `.offcanvas-title` class ensures proper margins and line-height. Then, add your content inside the `.offcanvas-body` class. To open the offcanvas sidebar, you must use a `<button>` or an `<a>` element that points to the id of the `.offcanvas` container (`#demo` in our example).

To open the offcanvas sidebar with an `<a>` element, you can point to `#demo` with the `href` attribute, instead of `data-bs-target` attribute.

```
<div class="offcanvas offcanvas-end" id="demo">
  <div class="offcanvas-header">
    <h1 class="offcanvas-title">Deloitte.com</h1>
    <button type="button" class="btn-close" data-bs-
dismiss="offcanvas"></button>
  </div>
  <div class="offcanvas-body">
    <p>Help Center</p>
    <p>Post a job</p>
    <p>Available jobs</p>
    <button class="btn btn-secondary" type="button">Sign Up</button>
  </div>
</div>

<div class="container-fluid mt-3">
  <h3>Right Offcanvas</h3>
  <button class="btn btn-primary" type="button" data-bs-toggle="offcanvas"
data-bs-target="#demo">
    Toggle Right Offcanvas
  </button>
</div>
```

Output:

