HTML(Hyper Text Markup Language)
=====================================

> HTML (Hypertext Markup Language) IS describes the structure of Web pages.

> With HTML you can create your own Web site.

> HTML is not a programming language.

> Markup languages are designed for the processing,definition and presentation of text.

Note:
> HTML is the only language that Browser understands.
> HTML is error free language.

Structure of html
=====================

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

       <title>Title of the document</title>
    </head>

    <body>
    Content of the document......
    </body>
</html>
```

Note: anything which is written inside <> we call as tag.

# <!DOCTYPE html>
    declaration defines this document to be HTML5.

# <html>
    it is the root tag of an HTML page.

# <head>
 tag contains meta information about the document.

# <title>
    tag specifies a title for the document.

# <body>
 tag contains the visible page content.

The main parts of our element are:
==================================

> The opening tag:
==================

This consists of the name of the element wrapped in opening and closing angle brackets(<>).

> The closing tag:
==================

```
This is the same as the opening tag, except that it includes a forward slash before
 the element name.

> The content:
===============
 This is the content of the element, which in this case is just text.

> The element:
================
The opening tag plus the closing tag plus the
 content equals the element.


HTML Basic Tags
=================
# <h1> to <h6>: Defines HTML headings
# <p>:  Defines a paragraph
        <p>this is paragraph</p>
# <br>: Inserts a single line break
# <hr>: to draw horizontal line
# <!--This is a comment-->: Defines a comment
# <b>:  Defines bold text
# <strong>: Defines important text
# <u>:  underlines the content
# <i>: defines in italics.
# <em>: Defines emphasized text
# <pre>:    Defines preformatted text.
# <sub>:    Defines subscripted text
# <sup>:    Defines superscripted text
# <mark>:Marked text
# <del>:Deleted text
# <small> - Small text
# <marquee>: it is used for scrolling piece of text.
# <bdo> element defines bi-directional override.

character Entities
===================
     non-breaking space              &nbsp;
<    less than                       &lt;
>    greater than                    &gt;
&    ampersand                       &amp;
"    double quotation mark           &quot;
'    single quotation mark           &apos;
¢    cent                            &cent;
£    pound                           &pound;
¥    yen                             &yen;
€    euro                            &euro;
©    copyright                       &copy;
®    registered trademark            &reg;
♥    black heart                     &hearts;
♦    black diamond                   &diams;
♣    black clubs                     &clubs;
♠    black spade                     &spades;

====================================================
∀       &forall;    FOR ALL
∂       &part;      PARTIAL DIFFERENTIAL
∇       &nabla;     NABLA
```

```
∏       &prod;      N-ARY PRODUCT
∑       &sum;       N-ARY SUMMATION
™       &trade;     TRADEMARK
←       &larr;      LEFTWARDS ARROW
↑       &uarr;      UPWARDS ARROW
→       &rarr;      RIGHTWARDS ARROW
↓       &darr;      DOWNWARDS ARROW
Α       &Alpha;     GREEK CAPITAL LETTER ALPHA
Β       &Beta;      GREEK CAPITAL LETTER BETA
Γ       &Gamma;     GREEK CAPITAL LETTER GAMMA
Δ       &Delta;     GREEK CAPITAL LETTER DELTA
Ε       &Epsilon;   GREEK CAPITAL LETTER EPSILON
Ζ       &Zeta;      GREEK CAPITAL LETTER ZETA
====================================================

Attributes:
==========
contain extra information about the element.

An attribute should always have:
===============================
A space between it and the element name.
The attribute name, followed by an equals sign.
Opening and closing quote marks wrapped around the attribute value.

EX:<a href="filepath"> HELLO</a>
====
Lists
=====
>There are two types of list
    -Ordered Lists
    -Unordered Lists

Ordered list (<ol>)
=====================
# <ol> element represents an ordered list of items.
# also called as Numbered Lists.

type:-
Indicates the numbering type:
        'a' indicates lowercase letters,
        'A' indicates uppercase letters,
        'i' indicates lowercase Roman numerals,
        'I' indicates uppercase Roman numerals,
        and'1' indicates numbers (default).

example:
========
<ol>
  <li>first item</li>
  <li>second item</li>
  <li>third item</li>
</ol>
Unordered Lists(<ul>)
=====================
> <ul> element represents an unordered list of items,
> also called as  bulleted list.

type:-
```

```
Indicates the bulleting type:
    -circle,
    -disc,
    -square.
example:
========
<ul>
  <li>first item</li>
  <li>second item</li>
  <li>third item</li>
</ul>

Image
======
->You can insert any image in your web page by using "<img>" tag.
->it can contain only list of attributes and it has no closing tag.
syntax
=======
<img src = "Image URL" ... attributes-list/>
Example
========
<img src="img.jpg" alt="Smiley face" height="42"
           width="42" align="right">
Marquee
=======
An HTML marquee is a scrolling piece of text displayed either
horizontally across or vertically down your webpage depending
on the settings.

Note
=====
The <marquee> tag deprecated in HTML5.

Syntax
======
<marquee>text message or image</marquee>

Attribute & Description
=======================
direction="value"
 This value can be:-up, down, left or right.

width="value"
 This value can be:-10 or 20%.

Links
============
->links that take you directly to other pages .
->These links are known as hyperlinks.
-> We can create hyperlinks using text or images available
->When you move the mouse over a link, the mouse arrow will
  turn into a little hand.
->A link is specified using HTML tag "<a>"
->This tag is called "anchor tag "
->anything between the opening <a> tag and the closing </a>
tag becomes part of the link
syntax
=======
<a href = "give URL">Text to Click</a>
```

```
Image Links
===========
 <a href = "Give Url">
         <img src = "image">
 </a>


Tables
======
-> tables are created using the "<table>" tag.
->"<th>" tag is used to create table heading.
-> "<tr>" tag is used to create table rows.
->"<td>" tag is used to create data cells.
<table border = "1">
         <tr>
            <th>Items</th>
            <th>Price</th>
         </tr>
         <tr>
            <td>1stRow  1stcolum</td>
            <td>1stRow, 2ndColum</td>
         </tr>
         <tr>
            <td>2ndRow, 1stColumn</td>
            <td>2ndRow , 2ndColumn </td>
         </tr>
 </table>
Cellpadding
=======================
->The cellpadding attribute specifies the space, in pixels,
between the cell wall and the cell content.

Eample
--------
<table cellpadding="2">
Cellspacing
=======================
->The cellspacing attribute specifies the space, in pixels,
between cells.
Note: Do not confuse this with the cellpadding attribute,
which specifies the space between the cell wall and the cell
 content.
Example
==========
 <table cellspacing="3">
 <table border = "1" cellpadding = "5" cellspacing = "5">
         <tr>
            <th>Name</th>
            <th>Salary</th>
         </tr>
</table>
Colspan and Rowspan
=====================
->colspan attribute use to merge two or more columns into a
single column.
->rowspan attribute use to merge two or more rows.
Example
==========
<table border = "1" bordercolor = "red" bgcolor = "blue">
```

```
          <tr>
             <th>1stColumn</th>
             <th>2ndColumn</th>
             <th>3Column</th>
          </tr>
          <tr>
             <td rowspan = "2">1stRow 1stCell </td>
             <td>1stRow 2ndCell</td>
             <td>1stRow  3rdCell</td>
          </tr>
          <tr>
             <td colspan = "3">Row 3 Cell 1</td>
          </tr>
       </table>
```

Table Height and Width
==================

Ex= <table border = "1" width = "100" height = "120">

Caption
==========

->The caption tag will serve as a title of table
->This tag is deprecated in newer version of HTML/XHTML.

Example
=========

```
 <table border = "1" width = "100%">
         <caption>Time table</caption>
```

Forms
=======

HTML forms are a very powerful tool for interacting with users;
"<form>" element is used to collect user input:

```
<form>
form elements
</form>
```

form Element
============

"<input>" element is the most important form element.

->we can displayed in several ways, depending on the "type" attribute.

Example
========

Type	Description
<input type="text">	Defines a one-line text input field.
<input type="radio">	Allowing a single value to be selected out of multiple choices.
<input type="checkbox">	Defines a checkbox.select ZERO or MORE options.
<input type="submit">	Defines a submit button.

"<select>" Element is used to drop down list.

=========

<option>" elements defines an option that can be selected.
->To define a pre-selected option, add the "selected"attribute to the option:
->Use the "size"attribute to specify the number of visible values.
-> to select more than one value use "multiple" attribute.

____
"<textarea>" Element represents a multi-line plain-text editing control.
=============
->"rows" attribute specifies the visible number of lines in a text area.
->"cols" attribute specifies the visible width of a text area.

____

"<fieldset>" element is used to group related data in a form.

"<legend>" element defines a caption for the <fieldset> element.

____

"<button>" element defines a clickable button:

Example
=========
```
<body>
<form>
   <fieldset>
    <legend>Personal information:</legend>
    First name:<br>
    <input type="text" name="firstname" value="Fname"><br>
    Last name:<br>
    <input type="text" name="lastname" value="Lname"><br><br>
  </fieldset>

  <input type="radio" name="institute" value="jspider" checked>
  jspider<br>
  <input type="radio" name="institute" value="Qspiders"> Qspiders<br>
   <input type="checkbox" name="Sub1" value="java"> I done java<br>
  <input type="checkbox" name="Sub2" value="j2ee"> I done j2ee

  <select name="text" size="2" multiple>
  <option value="value1">Value 1</option>
  <option value="value2" selected>Value 2</option>
  <option value="value3">Value 3</option>
  Hold down the Ctrl (windows) button to select multiple options.
</select>

<textarea name="textarea" rows="10" cols="30">
Write something here.
</textarea>

<button type="button" onclick="alert('Hello Shekar!')">Click
Me!</button>

<input type="submit" value="Submit">
<input type="reset" value="Reset">
</form>
</body>
```
==

```
File Upload Box
===============
->If you want to allow a user to upload a file to your web site.
->This is also created using the "<input>" element but
type attribute is set to file.
Example
========
<input type = "file" name = "fileupload" accept = "image/*" />
name:->Used to give a name to the control which is sent to the
 server to be recognized and get the value.
accept:->Specifies the types of files that the server accepts.
__________________________________________________________________________________
<input type="color"> is used for input fields that should
 contain a color.

Note: type="color" is not supported in Internet Explorer 11
and earlier versions or Safari 9.1 and earlier versions.

Example
=======
<form>
  Select your favorite color:
  <input type="color" name="favcolor">
</form>
__________________________________________________________________________________

"<input type="date"> "is used for input fields that should
contain a date.
Note: type="date" is not supported in Internet Explorer 11
and earlier versions.
Example
=======
<form>
Birthday:
<input type="date" name="bday">
</form>
__________________________________________________________________________________
"<input type="datetime-local"> "specifies a date and time input field,

Note: type="datetime-local" is not supported in Firefox,
or Internet Explorer 12 and earlier versions.
Example
=======
Joiningday (date and time):
  <input type="datetime-local" name="jointime">
__________________________________________________________________________________
<input type="email">is used for input fields that should
contain an e-mail address.
Example
========
E-mail:
<input type="email" name="email">
__________________________________________________________________________________
<input type="month"> allows the user to select a month and year.
```

Note: type="month" is not supported in Firefox, or Internet Explorer 11 and earlier versions.

Example

=======

```
Joiningday(month and year):
<input type="month" name="jdaymonth">
```

___

<input type="number"> defines a numeric input field.

We can set restrictions on what numbers are accepted.

Example

=======

```
 Quantity (between 1 and 10):
 <input type="number" name="quantity" min="1" max="10">
 Quantity:
 <input type="number" name="points" min="0" max="100" step="10"
 value="30">
```

___

<input type="range"> defines a control for entering a number whose exact value is not important.

Example

=======

```
<input type="range" name="points" min="0" max="10">
```

___

<input type="search"> is used for search fields.

->a search field behaves like a regular text field

Example

=======

```
Search Google:
  <input type="search" name="googlesearch">
```

___

<input type="tel"> is used for input fields that should contain a telephone number.

Note: type="tel" is only supported in Safari 8 and newer versions.

Example

=======

```
Telephone:
  <input type="tel" name="usrtel">
```

___

<input type="time"> allows the user to select a time (no time zone).

Example

=======

```
Select a time:
  <input type="time" name="usr_time">
```

___

<input type="url"> is used for input fields that should contain a URL address.

Example

```
=======
 Add your homepage:
  <input type="url" name="homepage">
______________________________________________________________________
__________
<input type="week"> allows the user to select a week and year.
Note: type="week" is not supported in Firefox, or Internet
Explorer 11 and earlier versions.
Example
=======
Select a week:
  <input type="week" name="year_week">
______________________________________________________________________
__________
"size" attribute specifies the size for the input
 field(in characters).
First name:<br>
Ex:<input type="text" name="firstname" value="shekar Gowda"
size="40">
______________________________________________________________________
__________
"maxlength" attribute specifies the maximum  length for the
 input field:
Ex:<input type="text" name="firstname" maxlength="15">
______________________________________________________________________
__________
 "height" and "width" attributes specify the height and width of
 an <input type="image"> element.
 Ex:<input type="image" src="a.gpg" alt="Submit" width="48"
 height="48">
______________________________________________________________________
__________
"placeholder"attribute specifies a hint that describes the
expected value of an input field
->The hint is displayed in the input field before the user
 enters a value.
-> its works with the following input types: text, search,
url, tel, email, and password.
Ex:<input type="text" name="fname" placeholder="First name">
______________________________________________________________________
__________
"required" attribute specifies that an input field must be
 filled out before submitting the form.
-> its works with the following input types: text, search,
url, tel, email, password, date pickers, number, checkbox,
radio, and file.
Ex: Username: <input type="text" name="usrname" required>
______________________________________________________________________
__________
"step"attribute specifies the legal number intervals for an
<input> element
-> its works with the following input types: number, range,
date, datetime-local, month, time and week



Attribute   Options Function
=============================
align
```

```
=====
right, left, center Horizontally aligns tags

valign
=====
top, middle, bottom Vertically aligns tags within
an HTML element.
bgcolor
======
numeric, hexidecimal, RGB values Places a background
 color behind an element
background:-URL Places a background image behind an element
id
====
User Defined    Names an element for use with
Cascading Style Sheets.
class
=====
User Defined Classifies an element for use
with Cascading Style Sheets.
width
=====
Numeric Value   Specifies the width of tables,
images, or table cells.
height
======
Numeric Value   Specifies the height of tables,
images, or table cells.
title:- User Defined"Pop-up" title of the elements.


===========================================================================================================
<audio> files could be played in a browser
Example
======
<audio src="path /good_enough.mp3" controls>
</audio>
       OR
<audio controls>
  <source src="aa.mp3" type="audio/mpeg">
</audio>
-> "controls" attribute adds audio controls, like play, pause, and volume.
"source"Defines multiple media resources for media elements, such as <video> and <audio>.
video
======
<video src="path pass-countdown.mp4" width="170" height="85" controls>
</video>
            OR
<video width="320" height="240" controls>
  <source src="movie.mp4" type="video/mp4">
  <source src="movie.ogg" type="video/ogg">
</video>
iframe
=======
<iframe> tag is used to specify an inline frame
Example
```

```
========
<iframe src="/html_iframe_tag_example.html" name="iframe_1" width="150" height="150"></iframe>

"src" Location of the frame contents OR  URL (web address) of the inline frame page.

"name"Assigns a name to a frame. This is useful for loading contents into one frame from
another.

"width" "height" Specifies the width and height of the inline frame.

=======================================================================
CSS
======
CSS stands for Cascading Style Sheets
=====================================
>CSS describes how HTML elements are to be displayed on screen.
>CSS is used to define styles for your web pages, including the design, layout.

CSS Syntax
==========
>CSS rule-set consists of a "selector" and a "declaration" block:
>p
{
 color: red;
 text-align: center;
}

> selector points to the HTML element you want to style.
>declaration block contains one or more declarations separated by semicolons.
>declaration blocks are surrounded by curly braces . declaration always ends with a semicolon.
Example
=======
h1
 {
  color: blue;
  background-color: yellow;
  border: 1px solid black;
}
p
{
  color: red;
}

Selectors
=============
>selectors are used to  select HTML elements based on their element name, id, class,
attributes.
>selector selects elements based on the "element name".
Example
=======
p
{
   color: red;
}
______________________________________________________________________________________
"id selector"uses the id attribute of an HTML element to select a specific element.
>id of an element should be unique within a page, so the id selector is used to select one
unique element.
>To select an element with a specific id, write a hash (#) character,
followed by the id of the element.
Note: An id name cannot start with a number.
Example
=======
#fid1
{
    text-align: center;
    color: red;
}

class Selector
==============
>class selector selects elements with a specific class attribute.
>To select elements with a specific class, write a period (.) character
followed by the name of the class
Note: A class name cannot start with a number!
Example
=======
.center
{
    text-align: center;
    color: red;
}
You can also specify that only specific HTML elements should be affected by a class.
p.center
{
    text-align: center;
    color: red;
}
Grouping Selectors
==================
>If you have elements with the same style definitions
>It will be better to group the selectors, to minimize the code.

how to Insert CSS to HTML ?
============================
3 different ways to apply CSS to an HTML.

1)Internal style sheet.

      2)Inline style sheet.

3)External style sheet.
______________________________________________________________________________________
1)Internal style sheet.
======================
```

>An internal style sheet may be used if one single page
>Internal styles are defined within the <style> element,
 inside the <head> section of an HTML page:

Example
=======

```
<html>
<head>
 <style>
h1
{
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

____

Note: Do not add a space between the property value and unit
->Like margin-left: 20 px; (its not works).
->The correct way is: margin-left: 20px;

____

 2)Inline style sheet.
===========
>An inline style may be used to apply a unique style for a single element
>add the style attribute to the relevant element.
>The style attribute can contain any CSS property.
Example
=========

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

____

3)External Style Sheet
======================
>With an external style sheet, you can change the look of an entire website by changing
just one file!
>Each page must include a reference to the external style sheet file inside the
<link> element.
The <link> element goes inside the <head> section:
>style sheet file must be saved with a .css extension.
>The file should not contain any html tags.
Example
=======

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

cssFile.css

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

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


colors
=======
Colors are specified using predefined
color names, OR
 RGB,OR
 HEX,OR
 HSL,OR
 RGBA,OR
 HSLA values.
using names
===========
Example
========
<h1 style="color:Tomato;">Hello World</h1>
<h1 style="background-color:DodgerBlue;">Hello HTML</h1>
<h1 style="border:2px solid Violet;">Hello CSS</h1>
________________________________________________________________________________________
Using RGB values
================
rgb(red, green, blue).
parameter  defines the intensity of the
color between 0 and 255.
Example
=======
<h1 style="background-color:rgb(255, 99, 71);">Shekar</h1>
________________________________________________________________________________________
using a hexadecimal value(HEX)
==============================
#rrggbb;
Where rr (red), gg (green), bb (blue) are hexadecimal values
between 00 and ff (same as decimal 0-255).

Example
=======
<h1 style="background-color:#ff0000;">my Name is Red</h1>
________________________________________________________________________________________
HSL Value
=========
hue, saturation, and lightness (HSL)
hue:->Green, orange, yellow, and blue — each of these is a hue,

Saturation:->  is also a percentage. 0% means a shade of gray,
and 100% is the full color.
```

```
>Saturation can be describe as the intensity of a color.
100% is pure color, no shades of gray
50% is 50% gray, but you can still see the color.
0% is completely gray, you can no longer see the color.

Lightness:-> is also a percentage.0% is black, 50% is neither light or dark,
100% is white
>lightness of a color can be described as how
much light you want to give the color,

Example
=======
<h1 style="background-color:hsl(147, 50%, 47%);"> my HSL color</h1>

________________________________________________________________________________
RGBA Value
===========
>RGBA color values are an extension of RGB color values with an alpha channel
>An RGBA color value is specified with:

rgba(red, green, blue, alpha)
> alpha parameter is a number between 0.0 (fully transparent)
and 1.0 (not transparent at all):
Example
=======
<h1 style="background-color:rgba(255, 99, 71, 0.4);"> my color is RGBA</h1>
________________________________________________________________________________
HSLA Value
==========
>HSLA color values are an extension of HSL color values with
an alpha channel - which specifies the opacity for a color.
HSLA color value is specified with:

hsla(hue, saturation, lightness, alpha)
Example
=======
<h1 style="background-color:hsla(9, 100%, 64%, 0.8);">my color is hsla</h1>
________________________________________________________________________________


Backgrounds
===========
CSS background properties:

background-color
background-image
background-repeat
background-attachment
background-position
________________________________________________________________________________
background-color: specifies the background color of an element.
=================
Example
=======
body
{
    background-color: lightblue;
}
______________________________________________________________________________________
Background Image
================
>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.
>By default, the background-image property repeats an image both horizontally and vertically.

Example
=======
body
{
background-image: url("imgpath.jpg");

}
>TO repeat an image harizontally.
>set background-repeat: repeat-x;

>To repeat an image vertically,
>set background-repeat: repeat-y;

>Showing the background image only once;
>background-repeat: no-repeat;
attachment
______________________________________________________________________________________

>position of the image is specified by the background-position property:
Example
=======
background-position: right top;

Background Image - Fixed position
=================================
Example
=======
background-attachment: fixed;


Shorthand property
===================
body {
    background: #ffffff url("img.png") no-repeat right top;
}
Multiple Backgrounds
====================
#example1
{
    background-image: url(img_flwr.gif), url(paper.gif);
    background-position: right bottom, left top;
    background-repeat: no-repeat, repeat;
}

shorthand property:
===================
Example
=======
#example1
{
    background: url(img_flwr.gif) right bottom no-repeat, url(paper.gif) left top repeat;
}
Background Size
===============
to specify the size of background images.
The two other possible values for background-size are contain and cover.

contain keyword scales the background image to be as large as possible
cover keyword scales the background image so that the content area is completely
covered by the background image.
#div1
{
       background-size: contain;
}
#div2
{

    background-size: cover;
}
Borders
=======
>border properties allow you to specify the
 style, width, and color
Border Style
=============
Example
======
 <style>
.solid {border-style: solid;}
.dotted {border-style: dotted;}
.dashed {border-style: dashed;}
.inset {border-style: inset;}
.outset {border-style: outset;}
.double {border-style: double;}
.groove {border-style: groove;}
.ridge {border-style: ridge;}
.none {border-style: none;}
.hidden {border-style: hidden;}
.mix {border-style: dotted dashed solid double;}
</style>

 ______________________________________________________________________________
border-width
============
border-width property can have from one to four values,
( top border, right border, bottom border,left border).
Example
=======
p{
    border-style: solid;
    border-width: 3px 4px 5px 150px;
}
__________________________________________________________________________________
Border Color
============
border-color property can have from one to four values ,
(top border, right border, bottom border, left border).
Example
========
p{
    border-style: solid;
    border-color: red green blue yellow;
}
__________________________________________________________________________________
border sides
=============
there are also properties for specifying each of the borders
(top, right, bottom, and left):
Example
=======
p {
    border-top-style: dotted;
    border-right-style: solid;
    border-bottom-style: dotted;
    border-left-style: solid;
}
__________________________________________________________________________________
Shorthand Property
==================
shorthand property for the following individual border properties:
>border-width
>border-style (required)
>border-color
Example
=======
p{
    border: 5px solid red;
}
__________________________________________________________________________________
we can  specify all the individual border properties for just one
side:
Example
=======
p
{
    border-left: 6px solid red;
    background-color: lightgrey;
}
__________________________________________________________________________________
```

border-radius property is used to add rounded borders to an element:
Example
========
```
p
{
    border: 2px solid red;
    border-radius: 5px;
}
```
____

Margins
========
margin properties are used to create space around elements,
outside of any defined borders.

Property	Description
margin-bottom	Sets the bottom margin of an element
margin-left	Sets the left margin of an element
margin-right	Sets the right margin of an element
margin-top	Sets the top margin of an element

Example
=======
```
p
{
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 80px;
}
```
Shorthand Property
==================
```
p
{
    margin: 25px 50px 75px 100px;
}
```
____
padding
=======
padding properties are used to generate space around an element's content,
inside of any defined borders.

Property	Description
padding-top	Sets the top padding of an element.
padding-right	Sets the right padding of an element.
padding-bottom	Sets the bottom padding of an element.
padding-left	Sets the left padding of an element.

Example
=======
```
div
{
    padding-top: 50px;
    padding-right: 30px;
    padding-bottom: 50px;
    padding-left: 80px;
}
```
Shorthand Property

```
==================
div
{
    padding: 25px 50px 75px 100px;
}
________________________________________________________________________________
________________________________________________________________________________

Box Model
=========
>All HTML elements can be considered as box.
>"box model" is used when talking about design and layout.
>It consists of: margins, borders, padding, and the actual content.


           _______________________________
                          Margin
           _______________________________

                         Border
           _______________________________
                         Padding
           _______________________________
                          Content
           _______________________________

           _______________________________

          ________________________________|
Content - The content of the box, where text and images appear
Padding - Clears an area around the content. The padding is
transparent
Border - A border that goes around the padding and content
Margin - Clears an area outside the border. The margin is transparent

Example
=======
Example
div
{
    width: 300px;
    border: 25px solid green;
    padding: 25px;
    margin: 25px;
}
_________________________________________________________________________________
Text
====
h1
{
    color: green;
}
h1
{
    text-align: center;
}
h1
```

```
{
    text-decoration: overline;
}

h1
{
    text-transform:uppercase;
}

h1
{
    text-transform: capitalize;
}
p
{
    text-indent: 50px;
}
h1
{
    letter-spacing: 3px;
}
p
{
    direction: rtl;
}
h1
{
    word-spacing: 10px;
}
h1
{
    text-shadow: 3px 2px red;
}

______________________________________________________________________________________
links
=====
Example

a{
color:red;
}
styled differently depending on what state they are in
======================================================
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

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;
}

list in css
============
list-style-type property specifies the type of list item marker.
_________________________________________________________________________________
Example
=========
<head>
<style>
.au
{
    list-style-type: circle;
}

.bu
{
    list-style-type: square;
}

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

.ao
{
    list-style-type: lower-alpha;
}
</style>
</head>
<body>
<p>unordered lists:</p>
<ul class="au">
  <li>pani puri</li>
  <li>masala puri</li>
  <li>kali puri</li>
</ul>


<p> ordered lists:</p>
<ol class="ao">
  <li>java</li>
  <li>j2ee</li>
  <li>Web</li>
</ol>

</body>
</html>


Tables
=======
>border-collapse property sets table borders should be collapsed
into a single border.
Example
=======
table
{
border-collapse: collapse;
}
Horizontal Alignment
====================
text-align property sets the horizontal alignment
th
{
    text-align: left;
}
Vertical Alignment
===================
vertical-align property sets the vertical alignment
Example
========
td
{
    height: 50px;
    vertical-align:top;
}
______________________________________________________________________
__________
border-bottom property to <th> and <td> for horizontal dividers.
Example
=======
th, td
{
    border-bottom: 1px solid red;
}

Hoverable Table
================
>hover selector on <tr> to highlight table rows on mouse over.
tr:hover
{
background-color:green;
}
______________________________________________________________________
__________
use the nth-child() selector and add a background-color to all even
or odd table rows.
tr:nth-child(even)
{
background-color:yellow;
}



display Property
================
>display property is the most important CSS property.
>It specifies how the element is displayed.
>The default display value for most elements is block or inline.
Block-level Elements
=====================
block-level element always starts on a new line.takes up the full width available.
Example
=======
<div>
<h1>____<h6>
<p>
<form>
<header>
<footer>
Inline Elements
================
inline element does not start on a new line and only takes up as much width as necessary.
<span>
<a>
<img>
Example
=======
li
{
    display: inline;
}
__________________________________________________________________________________
Hiding an element can be done by setting the display property to none.
Example
=======
.hidden
{
    display: none;
}
__________________________________________________________________________________
visibility:hidden;
>also hides an element.

>The element will still take up the same space as before.
>The element will be hidden, but still affect the layout.
Example
=======
.hidden
{
    visibility: hidden;
}
__________________________________________________________________________________
position
```

```
========
position property specifies the type of positioning method used for an element.

We have five different position values.
=========================================
->static
->relative
->fixed
->absolute
->sticky
_____________________________________________________________________________________
>Static positioned elements are not affected by the top, bottom, left, right properties.
Example
=======
div
 {
    position: static;
    border: 3px solid #73AD21;
 }


Combinators
===========
combinator is something that explains the relationship between the selectors.
There are four different combinators in CS
==========================================
1)descendant selector (space)
2)child selector (>)
3)adjacent sibling selector (+)
4)general sibling selector (~)
_____________________________________________________________________________________
descendant selector
====================
matches all elements that are descendants of a specified element.
div p
{
    background-color: yellow;
}
____________________________________________________________________________________
child selector
selects all elements that are the immediate children of a specified element.
div > p
{
    background-color: yellow;
}

_____________________________________________________________________________________
adjacent sibling selector selects all elements that are the adjacent siblings of a
 specified element.
 and "adjacent" means "immediately following".
```

Sibling elements must have the same parent element.
```
div + p
{
    background-color: yellow;
}
```
____

general sibling selector selects all elements that are siblings of a specified element.
```
div ~ p
{
    background-color: yellow;
}
```

____

JavaScript
==========
>JavaScript is an object-based scripting language that is
lightweight.
>JavaScript to program the behavior of web pages
>JavaScript Can Change HTML Content
Why JavaScript is used
=========================
JavaScript is used to create interactive websites.
Example
=======
*Displaying clocks.
*Client-side validation.etc

JavaScript Example
==================
```
<script>

  document.write("Hello JavaScript")

</script>
```
>script tag specifies that we are using JavaScript.
>document.write() function is used to display dynamic content through JavaScript.

3 Places to put JavaScript code
===============================
Between the body tag of html
Between the head tag of html
In .js file (external javaScript)

Variable
========
local variable and global variable.
>local variable is declared inside block or function.
It is accessible within the function or block only.

```
function xyz()
```

```
{
var x=10;//local variable
}
```

>global variable is declared outside the function
 it can be declared inside any function and can be accessed from any
 function.

Data Types
===========
>JavaScript is a dynamic type language, means we don't need to
specify type of the variable.
>use var here to specify the data type.
>It can hold any type of values such as numbers, strings etc.
Example
=======

```
var a=420;//holding number
var b="Raju";//holding string
```

In javaScript there is no default values.
O/P=undefined.

Operators
=========
same operators are present in javaScript as same as a java
But only differance in Relational operator.
== only compares values

=== compares values + type

```
0 == false   // true
0 === false  // false, because they are of a different type
1 == "1"     // true, automatic type conversion for value only
1 === "1"    // false, because they are of a different type
null == undefined // true
null === undefined // false
'0' == false // true
'0' === false // false
```

typeof:->is operator which is used to indicate
the what data type we have used for a variable.
this is present in javaScript.not in java.
Example
=======

```
document.writeln(typeof(a));
document.writeln(typeof(b));
```

===|
Control Statement: ALL ARE SIMILAR TO JAVA.|
===
Functions
=========
Advantage

Code reusability.
Less coding.
Example
=======

```
<script>
function msg()
{
alert("hello! this is message");
}
</script>
<input type="button" onclick="msg()" value="call function"/>


Function Return Value.
======================
<script>
function getValue()
{
return "hello shekar How r u?";
}
</script>

JavaScript Objects
==================
>JavaScript is an object-based language.
>Everything is an object in JavaScript.

3 ways to create objects.
========================
1>By object literal

property and value is separated by :(colon).

emp={id:102,name:"shekar",salary:40000};

document.write(emp.id+" "+emp.name+" "+emp.salary);

2>By creating instance of Object directly(using new keyword).

<script>
    var emp=new Object();
    emp.id=143;
    emp.name="Shekar";
    emp.salary=50000;
    document.write(emp.id+" "+emp.name+" "+emp.salary);
</script>

3>By using an object constructor (using new keyword)

>create function with arguments.
>Each argument value can be assigned in the current object by using
this keyword.

<script>

function emp(id,name,salary)
    {
    this.id=id;
    this.name=name;
    this.salary=salary;
    }
e=new emp(103,"Shekar Gowda",30000);

```

```
document.write(e.id+" "+e.name+" "+e.salary);

</script>


JavaScript Array
================
3 ways to construct array in JavaScript.
1>By array literal.
<script>
    var emp=["Shekar","Raju","Kirshna"];
    for (i=0;i<emp.length;i++)
    {
       document.write(emp[i] + "<br/>");
    }
</script>
2>By creating instance of Array directly (using new keyword)
<script>
    var i;
    var emp = new Array();
    emp[0] = "Arun";
    emp[1] = "Harshit";
    emp[2] = "Akash";

    for (i=0;i<emp.length;i++)
    {
    document.write(emp[i] + "<br>");
    }
</script>
3>By using an Array constructor (using new keyword)
<script>
    var emp=new Array("Priya","shweta","Sumera");
    for (i=0;i<emp.length;i++)
    {
    document.write(emp[i] + "<br>");
    }
</script>


String
======
2 ways to create string in JavaScript
1>By string literal
var str="This is javaScript";

2>By string object (using new keyword)

var stringname=new String("hello javascript string");
document.write(stringname);

String Methods
==============
1>charAt(index)
var str="javascript";
document.write(str.charAt(2));

2>concat(str)
var s1="javascript ";
var s2="example";
var s3=s1.concat(s2);
document.write(s3);
```

3>indexOf(str)
JavaScript String indexOf(str) method returns
the index position of the given string.

```
var s1="javascript from jspider";
var n=s1.indexOf("from");
document.write(n);
O/P=11
```

4>lastIndexOf(str)
The JavaScript String lastIndexOf(str) method returns
the last index position of the given string.

```
var s1="javascript from java indexof";
var n=s1.lastIndexOf("java");
document.write(n);
O/P=16;
```

5>toLowerCase()
```
var s1="JavaScript toLowerCase";
var s2=s1.toLowerCase();
document.write(s2);
```

6>toUpperCase()
```
var s1="JavaScript toUpperCase";
var s2=s1.toUpperCase();
document.write(s2);
```

7>slice(beginIndex, endIndex)
In slice() method, beginIndex is inclusive and endIndex is exclusive.
```
var s1="abcdefgh";
var s2=s1.slice(2,5);
document.write(s2);
O/P=cde
```

8>trim()
trim() method removes leading and trailing whitespaces from the string.
```
var s1="   I am javascript trim    ";
var s2=s1.trim();
document.write(s2);
o/p=I am javascript trim
```

Date
=====
Date objects are created with the new Date() constructor.

4 ways of initiating a date:

1>new Date()
creates a new date object with the current date and time:

Examples
========
```
<script>

var d = new Date();
document.getElementById("demo").innerHTML = d;

</script>
```

2>new Date(milliseconds)
```
<script>
var d = new Date(100000000000);
document.getElementById("demo").innerHTML = d;
</script>
```

3>new Date(dateString)
```
<script>
var d = new Date("October 13, 2014 11:13:00");
document.getElementById("demo").innerHTML = d;

</script>
```
4>new Date(year, month, day, hours, minutes, seconds, milliseconds)
```
<script>
var d = new Date(99, 5, 24, 11, 33, 30, 0);
document.getElementById("demo").innerHTML = d;
</script>
```

```
<button onclick="document.getElementById('demo').innerHTML=Date()">The time is?</button>

<p id="demo"></p>


<button onclick="this.innerHTML=Date()">The time is?</button>
```

Random number
=============
Math.random()
returns a random number between 0 (inclusive), and 1 (exclusive).
>Math.random() always returns a number lower than 1.

Math.floor(Math.random() * 10);
returns a number between 0 and 9

```
Math.floor(Math.random() * 11);
returns a number between 0 and 10

Math.floor(Math.random() * 100);
returns a number between 0 and 99

Math.floor(Math.random() * 101);
returns a number between 0 and 100

Math.floor(Math.random() * 10) + 1;
returns a number between 1 and 10

Math.floor(Math.random() * 100) + 1;
returns a number between 1 and 100

Math.round(4.7);
// returns 5
Math.round(4.4);
 // returns 4
Math.pow(2, 2);
 // returns 64



Example ahdhwefwefrger
========================
<script>


document.write("charAt(2)<br/>");
var str="javascript";
document.write(str.charAt(2));
document.write("<br/>");


document.write("cancat<br/>");
var s1="javascript ";
var s2="example";
var s3=s1.concat(s2);
document.write(s3);
document.write("<br/>");

document.write("IndexOf(str)<br/>");
var s4="javascript from jspider";
var n=s4.indexOf("from");
document.write(n);
document.write("<br/>");


document.write("lastIndexOf(str)<br/>");
var s5="javascript from java indexof";
var n=s5.lastIndexOf("java");
document.write(n);
document.write("<br/>");

document.write("tolowercase()<br/>");
var s6="JavaScript toLowerCase";
var s7=s6.toLowerCase();
document.write(s7);
```

```
 document.write("<br/>");

document.write("toUppercase()<br/>");
var s8="JavaScript toUpperCase";
var s9=s8.toUpperCase();
document.write(s9);
document.write("<br/>");

document.write("slice()<br/>");
var s10="abcdefgh";
var s11=s10.slice(2,5);
document.write(s11);
document.write("<br/>");

document.write("trim()<br/>");

var s12="   I am javascript trim    ";
var s13=s12.trim();
document.write(s13);
document.write("<br/>");
</script>



Date
======
<script>
var d = new Date();
document.getElementById("demo").innerHTML = d;



document.write("2>new Date(milliseconds)<br/>")

var d1= new Date(100000000000);
document.getElementById("demo1").innerHTML = d1;


document.write("3>new Date(dateString)<br/>");

var d2 = new Date("October 13, 2014 11:13:00");
document.getElementById("demo2").innerHTML = d2;

document.write("4>new Date(year, month, day, hours, minutes, seconds, milliseconds)");

var d3 = new Date(99, 5, 24, 11, 33, 30, 0);
document.getElementById("demo3").innerHTML = d3;
</script>


Browser Object Model (BOM)
==========================
is used to interact with the browser.
The default object of browser is window.
window Object  is created automatically by the browser.

Methods of window object
========================
```

alert()   displays the alert box containing message with ok button.

```
function msg()
{
 alert("Hello Alert Box");
}
```

___

confirm()   displays the confirm dialog box containing message with ok and cancel button.

```
function msg()
{
    var v= confirm("Are u sure?");
    if(v==true)
    {
     alert("ok");
    }
    else
    {
    alert("cancel");
    }
 }
```

___

prompt()   displays a dialog box to get input from the user.

```
function msg()
{
var v= prompt("What is you age?");
alert("your age is "+v);
}
```

___

open()   opens the new window.

```
function msg()
{
open("http://www.jspiders.com");
}
```

___

setTimeout()   performs action after specified time like calling function,
evaluating expressions etc.

```
function msg()
{
setTimeout(
function()
{
  alert("Welcome Shekar After 2 seconds")
},2000);

}
```

___

screen object holds information of browser screen.
It can be used to display screen width, height, colorDepth, pixelDepth etc.

Document Object Model(DOM)
==========================
>document object represents the whole html document.

Methods of document object
==========================
We can access and change the contents of document by its methods.

write("string")	writes the given string on the doucment.
writeln("string")	writes the given string on the doucment with newline character at the end.
getElementById()	returns the element having the given id value.
getElementsByName()	returns all the elements having the given name value.
getElementsByClassName()	returns all the elements having the given class name.
getElementsByTagName()	returns all the elements having the given tag name.

Accessing field value by document object
==
```
<script>
function printname()
{
var name=document.form1.name.value;
alert("Welcome: "+name);
}
</script>

<form name="form1">
Enter Name:<input type="text" name="name"/>
<button onclick="printname()">GetValue</button>
</form>
```

document.getElementById() method
=================================
```
<script>
function cube()
{
var number=document.getElementById("number").value;
alert(number*number*number);
}
</script>
<form>
Enter No:<input type="text" id="number" name="number"/><br/>
<button  onclick="cube()">Cube</button>
</form>
```
document.getElementsByName()
=============================
method returns a collection of all elements in the document with the specified name as a NodeList object.

The NodeList object represents a collection of nodes.
The nodes can be accessed by index numbers. The index starts at 0.

```
function totalelements()
{
var allgenders=document.getElementsByName("gender");
alert("Total Genders:"+allgenders.length);
}
</script>
<form>
Male:<input type="radio" name="gender" value="male">
Female:<input type="radio" name="gender" value="female">

<button onclick="totalelements()">NoOfGenders<button/>

document.getElementsByTagName()
===============================
method returns all the element of specified tag name.
<script>
function counth1tag()
{
var totalh1=document.getElementsByTagName("h1");
alert("total h1 tags are: "+totalh1.length);
}
</script>
<h1>This is a h1</h1>
<h1>Here  getElementByTagName() method.</h1>
<h1>Let's see the simple example</h1>
<button onclick="counth1tag()">count h1 tag</button>
_________________________________________________________________________________________
innerHTML property can be used to write the dynamic html on the html document.
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML = "Hello World!";
</script>
getElementById is a method, while innerHTML is a property.
_________________________________________________________________________________________

Events
======
onblur
======
When you leave the input field, a function is triggered which transforms
the input text color to red.
<script>
function myFunction()
{
    var x = document.getElementById("fname");
    x.style.color ="red";
}
</script>

Enter your name: <input type="text" id="fname" onblur="myFunction()">
_________________________________________________________________________________________
onfocus
===========
```

When the input field gets focus, a function is triggered which changes the
background-color.

```
<script>
function myFunction(x)
{
    x.style.background = "yellow";
}
</script>
Enter your name: <input type="text" onfocus="myFunction(this)">
```

____

```
<script>
function myFunction()
{
    document.getElementById("demo").innerHTML = "You selected some text";
}
</script>
<body>

Some text: <input type="text" value="Hello world!" onselect="myFunction()">

<p id="demo"></p>
```

____

onsubmit
========

```
function confirmInput()
{
    fname = document.forms[0].fname.value;
    alert("Hello " + fname + "! You will now be redirected to www.w3Schools.com");
}
</script>

<form onsubmit="confirmInput()" action="https://www.jspiders.com/">
Enter your name: <input id="fname" type="text" size="20">
<input type="submit">
```

____

onkeydown
=========

A function is triggered when the user is pressing a key in the input field.

```
function myFunction()
{
    alert("You pressed a key inside the input field");
}
</script>

<input type="text" onkeydown="myFunction()">
```

____

onkeyup
=======

A function is triggered when the user releases a key in the input field.

```
The function transforms the character to upper case.
function myFunction()
{
    var x = document.getElementById("fname");
    x.value = x.value.toUpperCase();
}n m
</script>


Enter your name: <input type="text" id="fname" onkeyup="myFunction()">
_________________________________________________________________________________________
function color(color)
{
    document.forms[0].myInput.style.background = color;
}
</script>

<form>
Write a message:<br>
<input
type="text"
onkeydown="color('yellow')"
onkeyup="color('white')"
name="myInput">
</form>
_________________________________________________________________________________________

_________________________________________________________________________________________
Mouse Events
=============
onmouseover and onmouseout
==========================
<h1 onmouseover="style.color='red'"
onmouseout="style.color='black'">Mouse over this text</h1>
_________________________________________________________________________________________
onmousedown and onmouseup
==========================
<script>
function myFunction(element, color)
{
    element.style.color = color;
}
</script>
</head>
<body>

<p onmousedown="myFunction(this,'red')"
onmouseup="myFunction(this,'green')">
Click the text to change the color.
</body>
_________________________________________________________________________________________
<script>
function bigImg(x)
{
    x.style.height = "64px";
    x.style.width = "64px";
}
function normalImg(x)
{
    x.style.height = "32px";
    x.style.width = "32px";
}
</script>
<body>

<img onmouseover="bigImg(this)" onmouseout="normalImg(this)" border="0" src="imgsrc.jpg"
alt="" width="32" height="32">
</body>
_____________________________________________________________________________________
Form Validation
===============
validate the name and password. The name can’t be empty and
password can’t be less than 6 characters long.

function validateNameandPassform()
{
var name=document.myform.name.value;
var password=document.myform.password.value;

if (name==null || name=="")
{
  alert("Name can't be blank");
  return false;
}else if(password.length<6)
{
  alert("Password must be at least 6 characters long.");
  return false;
  }
}
</script>
<body>
<form name="myform" onsubmit="return validateNameandPassform()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password" name="password"><br/>
<input type="submit" value="register">
</form>
Password Validation
===================
<script>
function matchpass()
{
var firstpassword=document.f1.password.value;
var secondpassword=document.f1.password2.value;

if(firstpassword==secondpassword)
{
return true;
}
else
{
alert("password must be same!");
return false;
}
}
</script>

<form name="f1" onsubmit="return matchpass()">
Password:<input type="password" name="password" /><br/>
Re-enter Password:<input type="password" name="password2"/><br/>
<input type="submit">
</form>
________________________________________________________________________________
<script>
function validate()
{
var msg;
if(document.myForm.userPass.value.length>6)
{
msg="good";
}
else
{
msg="poor";
}
document.getElementById('mylocation').innerText=msg;
 }

</script>
<form name="myForm">
<input type="password" value="" name="userPass" onkeyup="validate()">
Strength:<span id="mylocation">no strength</span>
</form>
________________________________________________________________________________

Number Validation
=================
<script>
function validate()
{
    var num=document.myform.num.value;
    if (isNaN(num)){
      document.getElementById("num").innerHTML="Enter Numeric value only";
      return false;
    }
    else
    {
      return true;
    }
}
</script>
<form name="myform" onsubmit="return validate()" >
Number: <input type="text" name="num"><h1 id="num"></h1><br/>
<input type="submit" value="submit">
</form>
```

What Is a Regular Expression?
=============================
A regular expression is a sequence of characters that forms a search pattern.

Syntax
======
```
var reg= /pattern/modifier;
```

where "pattern" is the regular expression.
and the "modifiers" is optionals

Modifiers
==========

i	Perform case-insensitive matching
g	Perform a global match (find all matches rather than stopping after the first match)
m	Perform multiline matching

Brackets are used to find a range of characters:
===

Expression	Description
[a-z]	Find any character between the brackets
[0-9]	Find any character between the brackets (any digit)
(x|y)	Find any of the alternatives specified

Metacharacters are characters with a special meaning:
===

\d	Find a digit 0-9
\w	Find a all words and A-z a-z 0-9
\s	Find a whitespace character
\b	Find a match at the beginning or at the end of a word

___

Quantifiers define quantities:
==============================

n+	Matches any string that contains at least one n(1 or more)
n*	Matches any string that contains zero or more occurrences of n(0 or more)
n?	Matches any string that contains zero or one occurrences of n(0 or 1)

___

```
var regex = /^\d{2}$/;
```
The pattern portion above starts with an ^ indicating the beginning of a string.
The \d indicates a digit followed by {2} meaning 2 consecutive digits.
The $ indicates end of a string.
So,this pattern will attempt to find
exactly 2 consecutive digits from the beginning to the end of a string.

___

HTML form contains only letters.
================================
```
 var letters = /^[A-Za-z]+$/;
```

___

```
To get a string contains only numbers (0-9)
===========================================
var dig=/^[0-9]+$/
which allows only numbers.
________________________________________________________________________________
validate a phone number of 10 digits with no comma,
======================================================
var phoneno = /^\d{10}$/
permit only phone numbers with 10 digits.
________________________________________________________________________________
contains letters and numbers only
=========================================
var letterNumber = /^[0-9a-zA-Z]+$/
________________________________________________________________________________
whether an input string is a valid emai
========================================
/^[a-zA-Z0-9._-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,4}$/
      OR
(/^\w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/


Uppercase (A-Z) and lowercase (a-z) English letters.
Digits (0-9).
Characters ! # $ % & ' * + - / = ? ^ _ ` { | } ~
Character . ( period, dot or fullstop) provided that it is not the first or last
character and it will not come one after the other.
The domain name [for example com, org, net, in, us, info]
part contains letters, digits, hyphens, and dots.

Example of valid email id
=========================
mysite@ourearth.com
my.ownsite@ourearth.org
mysite@you.me.net

Example of invalid email id
============================

mysite.ourearth.com [@ is not present]
mysite@.com.my [ tld (Top Level domain) can not start with dot "." ]
@you.me.net [ No character before @ ]
mysite123@gmail.b [ ".b" is not a valid tld ]
mysite@.org.org [ tld can not start with dot "." ]
.mysite@mysite.org [ an email should not be start with "." ]
mysite()*@gmail.com [ here the regular expression only allows character, digit,
underscore, and dash ]
mysite..1234@yahoo.com [double dots are not allowed]

Character   Description
=========================
/ .. /  All regular expressions start and end with forward slashes.
________________________________________________________________________________
^   Matches the beginning of the string or line.
```

\w+ Matches one or more word characters including the underscore. Equivalent to [A-Za-z0-9_].

____

[\.-]   \ Indicates that the next character is special and not to be interpreted literally.
.- matches character . or -.

____

? Matches the previous character 0 or 1 time. Here previous character is [.-].

____

\w+ Matches 1 or more word characters including the underscore. Equivalent to [A-Za-z0-9_].

____

* Matches the previous character 0 or more times.

____

([.-]?\w+)* Matches 0 or more occurrences of [.-]?\w+.

____

\w+([.-]?\w+)* The sub-expression \w+([.-]?\w+)* is used to match the username in the email.

____

It begins with at least one or more word characters including the underscore, equivalent to [A-Za-z0-9_]. , followed by . or - and . or - must follow by a word character (A-Za-z0-9_).
@ It matches only @ character.

____

\w+([.-]?\w+)* It matches the domain name with the same pattern of user name described above.

____

\.\w{2,3} It matches a . followed by two or three word characters, e.g., .edu, .org, .com,
.uk, .us, .co etc.

____

+ The + sign specifies that the above sub-expression shall occur one or more times,
e.g., .com, .co.us, .edu.uk etc.

____

$ Matches the end of the string or line.

____

Note: If you want to work on 4 digit domain,
for example, .info then you must change w{2,3} to w{2,4}.

____

n{X}	Matches any string that contains a sequence of X n's
n{X,Y}	Matches any string that contains a sequence of X to Y n's
n{X,}	Matches any string that contains a sequence of at least X n's
n$	Matches any string with n at the end of it
^n	Matches any string with n at the beginning of it

?=n	Matches any string that is followed by a specific string n
?!n	Matches any string that is not followed by a specific string n