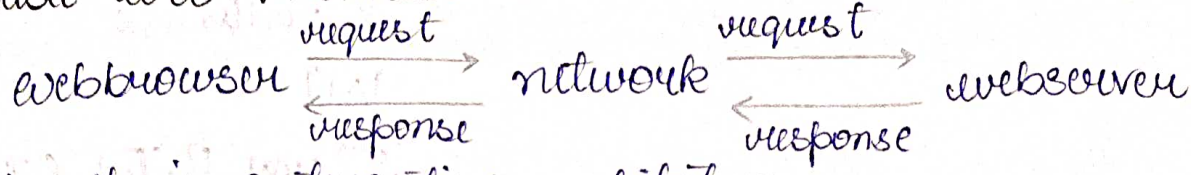
28/7/22

UNIT-2

Introduction to web

* Basic web Architecture :-



→ The web is a two-tier architecture.

→ Web server basic objective is to store, process and deliver web pages to the user.

→ This intercommunication is done using HTTP.
(Hypertext transfer protocol.)

* Client side Technologies :-
  - HTML, CSS, Javascript, VBScript.
  - XHTML, DHTML.

* Server side Technologies :-
  - ASP, PHP, Perl, JSP
  - ASP, NET, Java
  - MSQL, SQL server.

World wide web

In 1989, Tim Berners-Lee has developed URL'S, HTPP, HTML.

* URI :- (also called as URL)

→ URL's are location dependent.

→ http://www.weather.com/weather/us/zips/54701.html

- http → Communication protocol
- www.weather.com → webserver domain name
- weather/us/zips → folder path (Optional)
- 54701.html → web page. HTML file optional.

→ There are several kinds of URL'S
file URL'S, FTP URL, etc.

* HTTP :-

→ It is an application - level protocol for distributed, collaborative, hypermedia, information systems.
→ It is a request/response standard of a client & a server.
→ Typically, an HTTP client initiates a request.
→ Resources to be accessed by HTTP are identified using (URIs) Uniform Resource Identifiers.

HTML

→ stands for Hyper Text markup language.
→ standard markup language for creating web pages.
→ describes the structure of a webpage.
→ consists of a series of elements.
→ elements tell the browser how to display the content.

* Structure of HTML :-

```
<html>
    <head>
        <title>.....</title>
    </head>
    <body>
    </body>
</html>
```

→ <h1> element defines a large headings.
→ <p> element defines a paragraph.
→ Html element is defined by a start tag, some content, and an end tag.

<tagname> content goes here..... </tagname>

Eg:- <h1> My first heading </h1>
 My first paragraph

→ attribute gives additional information about ~~one~~ an element in HTML.

Eg:- <body background = "images/back.png">
<body bg colour = "#00ccee">
<text>

General form:-
<tagname attribute = "value">

→ attributes are specified in the start tag.

→ They usually comes in starting tag.

29/7/22

<B> Bold text
<I> Italic
<STRIKE> strike out text.

* Image tag:-

→ <img> is used to embed an image in a web page.

→ The <img> tag is empty, it contains attributes only.

→ does not have a closing tag.

→ contains 2 attributes
1) SRC → specifies path of image.
2) alt → specifies an alternate text for the image.

Syntax:-
<img SRC = "url" alt = "alternate text">

→ attributes
=> ALIGN => left or right
=> ALT => alternate text
=> BORDER => puts border around the image.
=> WIDTH
=> HEIGHT

Url ⇒ path where image is stored.

* List :- helps or allow web developers to group a set of related items in list.

1) unordered HTML list :-
   → starts with <ul> tag
      list item starts with <li> tag.
   → By default, bullets are marked.

Eg:- <ul type = "square">
     <ul type = "disc">
     <ul type = "circle">

→ type is the attribute of unordered list.

Eg: <body>
      <ul type = "square">
         JAVA
         Python
         WT

    </body>

2) ordered list :-
   → starts with <ol> tag

(i) <ol type = "A">

(ii) <ol type = "1" start = "4">

(iii) <ol type = "I"> .

3) discription list :-
→ <dl> tag defines the description list, the <dt> tag defines the item (name), & the <dd> tag describes each term.

Eg:- <dl>
       <dt> java </dt>
       <dd> java is oop </dd>
     </dl>

Assignment :-

1) A. Courses
   - I. Graduation courses
     - B.Tech
       1. CSE
       2. CSE
     - B.SC
       1. BSC-CS
       2. BSC - MPC
   - II. Post Graduation Courses
     - M.Tech
       1. CSE
       2. ECE
   - B. Placements
     1. CSE
     2. IT

2) Create Bio data by using Basic HTML tags.

3) design a web page in below format using list & basic tags.

Table :-

→ The HTML tables allow web authors to arrange data like text, images, links into rows & columns of cells.

<table> Create table
<tr> Create table rows
<td> Create data cells
<th> table header

Eg:
```
<table>
    <th>
        <td>....</td>
        <td>....</td>
    <th>
```

```
<tr>
    <td>.....</td>
    <td>.....</td>
</tr>
</table>
```

→ Attributes :-

1) BORDER (border for rows & columns).
2) Cell padding (Cell padding is the space between the cell edges & the cell content)
3) Cell spacing (Cell spacing is the space b/w the circle.
4) BG colour
5) Border colour
6) width = "PX"
7) height = "px"

<TR> → align = left, centre, right

<TD> → colspan = "number" (spans cells across column) ↗ merging two columns
Rowspan = "number" (spans cells across rows ↘ merging two rows.

<TH> → colspan = "number"
Rowspan = "number"

30/7/22

HTML Frames

→ It divides a browser window into several divisions, each division containing a separate HTML document.
→ A collection of frames in browser window is known as a frameset.
→ There are still few browsers who do not support frame technology.
→ <frameset> tag replaces the <body> element in frameset documents.

→ attributes : cols, rows

Eg :- cols = "100, 500, 100"

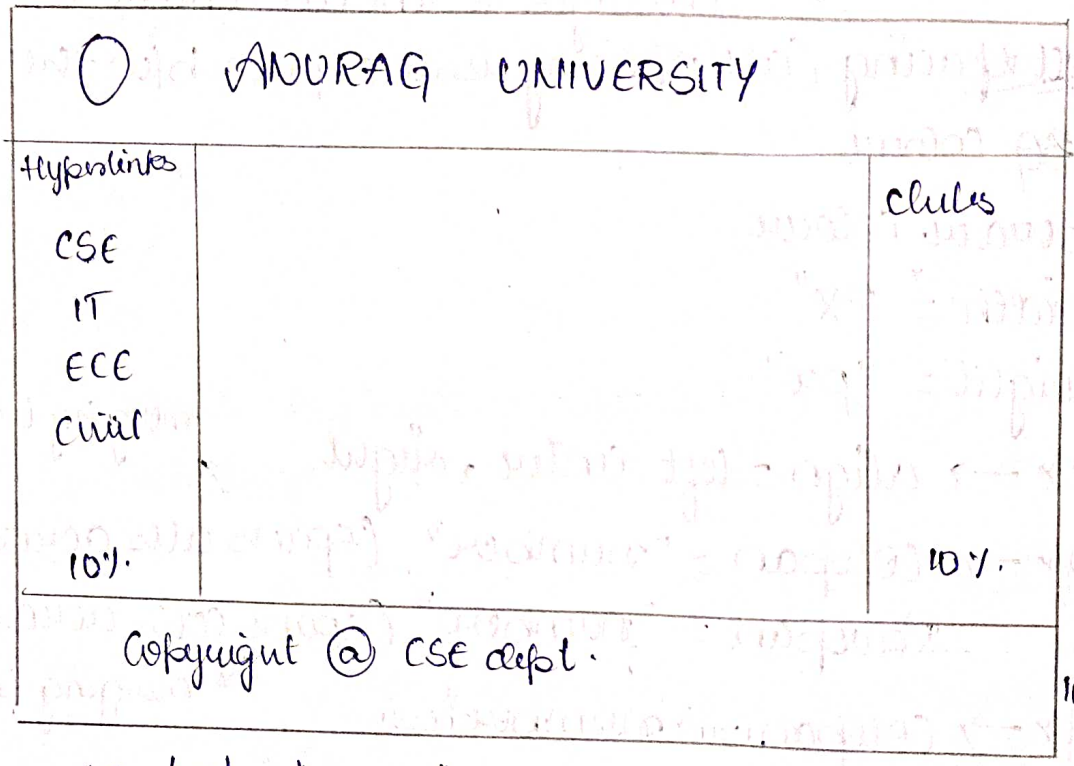
cols = "10%, 80%, 10%"

cols = "10%, *, 10%" same as cols = "10%, 80% 10%"

Eg : rows = "20%, *"

rows = "20%, 20%, 20%, 20%, *"

* Assignment :-



<a href = " ..... " >.

4/8/22

* Frameset attributes:-

⇒ rows :- specifies the rows in the frameset.

⇒ border :- specifies the width of the border of each frame in pixels.

⇒ frameborder :- specifies whether a three-dimensional border should be displayed b/w frames. this attribute takes value either 1 (yes) or 0 (no). Eg :- frameborder = "0" specifies no border.

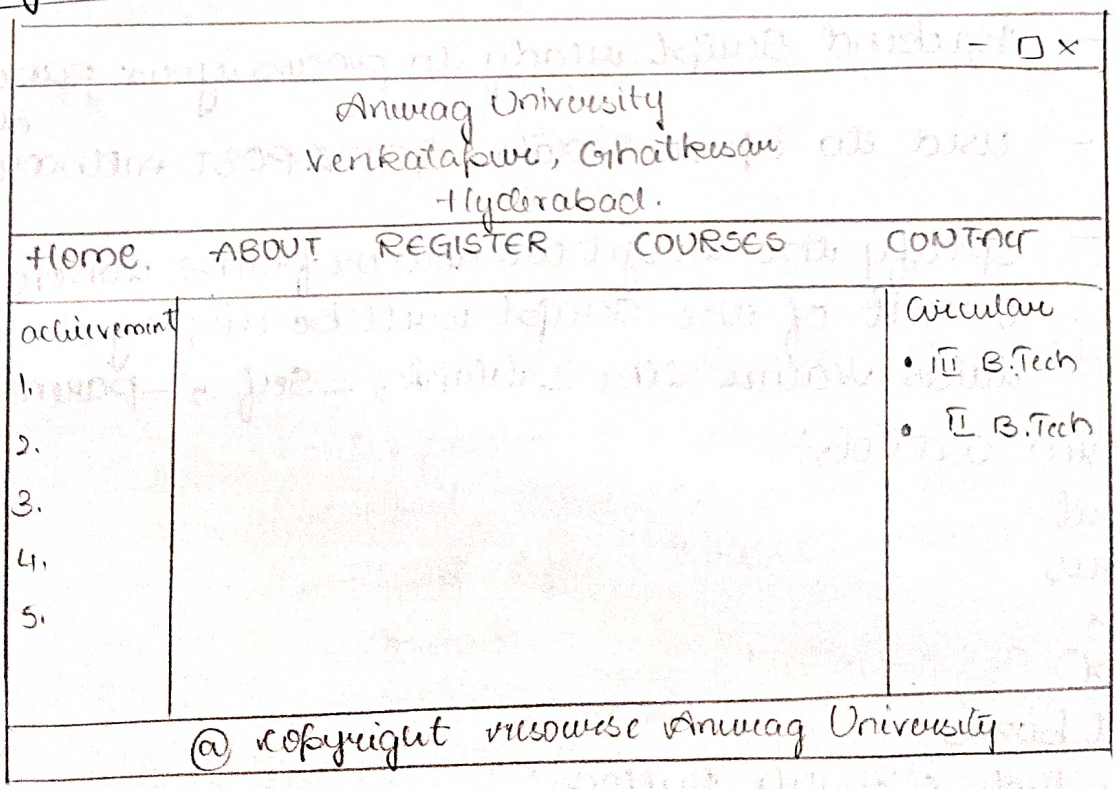
⇒ noresize :- Cannot resize the frame in the browser.
<frameset noresize = "no" > , <frameset noresize = "no" >

⇒ src :- indicates the frame file that should be used in the frame.

⇒ name :- gives name to frame.

⇒ Scrolling :- Eg :- Scrolling = "no" means it should not have Scroll bars.

Assignment :-



* noframe is used to display the error when there is no support from the browser.

HTML forms :-

⇒ A form will take input from the site visitor & then will post it to a back-end application such as ASP....

⇒ Elements in forms

1. text fields
2. text area fields
3. Drop-down menus
4. Radio buttons
5. Checkboxes
6. etc

Syntax :- <form action = "URL"
method = "GET/POST" >

</form>

* form attributes

1. action – Backend Script ready to process your passed data
2. method – used to upload data. GET & POST methods.
3. target – specify the target window or frame where the result of the script will be displayed. It takes values like _blank, _self, _parent etc.

* HTML form controls :-

1. Text input
2. Checkboxes
3. Radio Box
4. Select Box
5. file Select boxes
6. Hidden controls Clickable Buttons
7. Subit & Reset Button

5/8/22

* Types of input controls:-

single

1. Text input controls:-

1) Single-line text input controls => <input> tag
2) Password input => <input> tag
3) Multiline text input controls => <textarea> tag

Eg: 1.
```
<body>
<form>
firstname : <input type="text" name="first_name" />
</form>
</body>
```

=> attributes:

type => type of input control

name => name to control.

value => provide an initial value inside the control.

size => width of the text-input control in terms of characters.

maxlength => max no. of characters a user can enter into the textbox

Eg: 3.
```
<form>
<textarea rows="5" cols="50"
name="description">.....
</textarea>
</form>
```

*
```
<form action="#">
enter username : <input type="text"><br>
enter password : <input type="password"><br>
<input type="submit" name=" ">
<input type="reset" name=" ">
</form>
<textarea rows="10" cols="50">
Address.
</textarea>
```

2. Check box control

⇒ attributes:

type ⇒ <input type = "checkbox" > .

name ⇒ name to control

value ⇒ the value that is checked at the time of Text field.

Checked ⇒ set to checked if you want to select it by default.

⇒ more than one option is required.

3. Radio button

⇒ attributes

type

name

value

checked

Eg:

```
<body>
 <form>
<input type = "radio" name = "subject" value = "maths">
                                                  maths.
<input type = "radio" name = "Science" value = "physics">
 </form>
</body>.
```

⇒

first name : [ ]

last name : [ ]

user name : [ ]

password : [ ]

GENDER : O male O female

JOB : [ ]

[submit] [reset]

4. Select box Control :-

Eg:
```
<form>
<select name = "drop down" >
<option value = "maths" selected > maths </option >
<option value = "physics" > physics </option >
</select>
</form.
```

* It is also called drop down box which provides option to list down various options in the form of dropdown list.

=> attributes :-

name

size

multiple.

=> attributes of <option tag>

value

selected

label. => alternate way of labelling option.

CSS Selectors

* CSS selectors are used to select the content you want to style.

There are several different types of selectors in CSS

1) CSS Element Selector
2) CSS Id selector (#)
3) CSS class selector (.)
4) CSS Universal selector (*)
5) CSS Group selector (,)

1) Element Selector :-

Selects the HTML element by name.

Eg:
```
<head>
<style>
    p { text-align: center;
        color: blue;
    }
</style>
</head>
<body>
    <p> this is a paragraph </p>
    <p id = "para1"> Me+Ti !, </p>
    <p> And me! </p>
</body>
```

2) Id Selector :- (#)

The id selector selects the id attribute of an HTML element to select a specific element. An Id is always unique within the page so it is chosen to select a single, unique element.

Eg:
```
<style>
#para1 { text-align : center;
         Colour : blue;
       }
</style> </head>
<body> <p id = "para1"> Hello </p>
<p> welcome </p>
</body>.
```

3) Class selector :- Selects HTML elements with a specific class attribute. It is used with a period character . (full stop symbol) followed by the class name.

Eg:
```
<style>
.center{
    text-align:center;
    color:blue;
}
</style></head>
```

```
<body>
<h1 class="center">Hi</h1>
<p class="center">Hello</p>
</body>
```

4) Universal Selector :-
The universal selector is used as a wildcard character. It selects all the elements on the page.

```
<style>
*{
    color:green;
    font-size:20px;
}
</style></head>
```

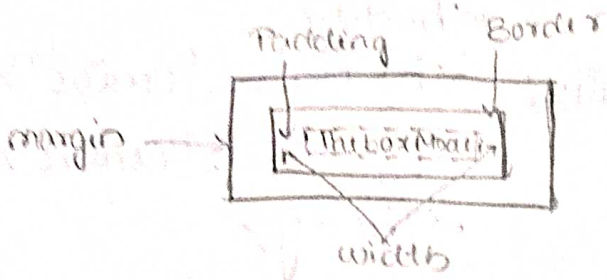
```
<body>
<h2>This is me</h2>
<p>Hi</p>
<p id="para3">Hello</p>
<p>welcome</p></body>
```

5) Group Selector :-
It is used to select all the elements with the same style definitions. It is used to minimize the code. Commas are used.

Eg:
```
h1,h2,p{
    text-align:center;
    color:blue;
}
```
(or)
```
h1{
    text-align:center;
    color:blue;
}
h2{
    text-align:center;
    color:blue;
}
p{
    text-align:Center;
    Color:blue;
}
```

17/6/22

* Padding :-
→ distance b/w content & Border.
→ Padding width : Adds to the total Box width.



* Borders :-
3 properties : width, style and color.
Border width adds to total Box width.

Eg:- border : 1px solid red;
(or)
borders = inset
outset
dotted
ridge
groom
hidden
none
} 3D border.

border-style : dashed
border-width : 5px
border-color : red

Eg:
border-top-style : ridge
border-bottom-style : solid
border-right-style : dashed.
border-left-style : inset
border-color

* Margins :
distance b/w one element & adjacent elements

* Box width :-
Box width = Content width
+ padding + right

* Box height :-
Box height = Content height

CSS - Pseudo classes :-

CSS- Pseudo classes are used to add special effects to some selectors.

- A simple syntax of pseudo-classes is as follows-

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

The :link pseudo-class:-

how to use :link class to set the link color.

```
<html>
 <head>
 <style type = "text/css">
 a:link {
    color : yellow;
 }
 </style> </head> <body>
 <a href = "#"> Email </a> </body>
 </html>.
```

Eg:

```
<html> <head> <style>
 a:link { color : yellow : } </style> </head>
 <body>
 <a href = "html 1"> click here </a>
 </body> </html>
```

The :visited pseudo-class:-

how to use the :visited class to set the color of visited links.

```
<html> <head>
<style type = "text/css">
 a: visited (color : #001 )
 </style>
```

* The :hover pseudo class:-

-: hover class to change the colour of links when we bring a mouse pointer over that link.

```
<head>
<style type="text/css">
a:hover {colour: green}
</style> </head>
<body> <a href=" "> click </a>
</body> </html>
```

* The :active pseudo-class:-

: active to change the colour of active links.

```
<style>
a: active {Colour: #FF00CC}
</style
```

Pseudo-element

⇒ It allow you to style the elements or parts of the elements without adding any ID's.

Syntax:

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

1) The ::first-line Pseudo-element:-

It applies special style to the first line of a text.

Eg:

```
<style>
    p::first-line {
        Colour: red; }
</style>
</head>
<body> <p>.
```

2) the ::first-letter Pseudo-element :-
used to add a special style to the first letter of the first line of a text.

Eg:
```
p::first-letter { color : #ff0000; font-size: xx-large; }
```

3) The ::before & ::after Pseudo-element :-
used to insert generated content either before or after an element's content.

Eg:
```
h1::before {
    content : url("images/marker-left.gif") or "text";
}
h1::after {
    content : url("images/marker-right.git") or "text";
}
```

```
* {
    margin : 0;   // no space b/w border & web page //.
}
```

=>
```
#menu {
    display : inline-block;
    background-color : grey;
    width : 100%;
    height : 60px;
}
```

=>
```
li a {
    text-decoration : none;
    padding : 30px;
    color : white;
}
```

=> # header { width = 100%; }

=> descendant element
```
ul li {
    list-style : none;
    float : left;
    padding-top : 18px;
    font-size : 20px;
}
```

=>
```
a:hover {
    background-color : yellow;
    border-radius : 10px;
}
```

24/8/22

Span & div

* Span tag is an inline tag. where as tag is used as block tag.

```
<div> 1 </div>
<div> 2 </div>
<div> 3 </div>
```

o/p
1
2
3

```
<span> 1 </span>
<span> 2 </span>
<span> 3 </span>
```

o/p 1 2 3.

Eg:
```
<h1> Anurag <span style = "color: red" >
        University </span> </h1>.

h1 { color : green; }.
```

o/p: Anurag (green.) University (red)

* Inline frames:-

<iframe> </iframe>.

=> attributes are name, height, width, border, src, etc.

=> There are two types of validation

Client side validation.

Server side validation

24/8/22

Introduction to javascript

→ It is used to create client-side dynamic pages.

→ It is OOP language which is used by several websites.

→ It is an interpreted that enables dynamic interactivity on websites.

* Features of JS:-

→ All web browsers support JS as they provide built-in execution environments.

→ JS follows the syntax & structure of the C programming lang.

→ It is a case-sensitive language.

→ JS is supportable in several OS. eg: Linux, windows, etc.

* Applications of JS:-

→ client-side validation.

→ Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm box & prompt dialog box).

⇒ Syntax:-

```
<Script type = "JavaScript/text">
    //code.
</Script>
```

Eg:-

```
<html>
  <body>
   <h2> Welcome to JS </h2>
   < Script >
    document.write ("Hello JS");
   </Script>
  </body> </html>
```

* The document.write() function is used to display dynamic content through JS

* Places to put JS code :-

1. B/w the body tag of html.
2. B/w the head tag of html.

Eg:
```
<head>
  <script>
  </script>
</head>.
```

* Types of Comment :-

1. Single line Comment (//)
2. multi-line Comment (/*....*/)

* Java Script Variable :-

A JS variable is simply a name of storage location.
There are two types of variables in JS : local variable and global variable.

There are some rules while declaring a JS variable :-

1) Name must start with a letter, underscore, dollar ($)
2) After first letter we can use digit Eg :- value1
3) JS variable are case sensitive, Eg :- x & X are different.

Eg:
```
<html>
  <body>
    <script>
      var x = 10;
      var y = 20;
      var z = x+y;
      document.write(z);
    </script> </body> </html>
```

* JS datatypes:-

Two types

1) Primitive datatype

2) Non-primitive datatype.

There are 5 primitive datatypes:-

String (characters)

Number (numeric values)

Boolean (True or false)

Null (no value at all.

26/8/22

* JS functions:-

Syntax:- function functionname (arguments)

{ //body

}

JS functions are used to perform operations.

Advantages:-

1. code reusability.

2. less code.

Eg:- (1)

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

(2)
```
<body>
<script>
function getcube(number)
{
  alert("Result =" + number * number * number);
}
</script>
<form>
 < input type = "button" onclick = "getcube(4")"
                          value = "Result" />
</form>
</body>.
```

* for displaying in table : replace alert with document.write.

(3)
```
<body>
<script>
function getcube(number)
{
return (number * number * number);
}
</script>
<form> <script>
  document.write(getcube(4));
</form> </script>
</body>
```

* JS function object :- (state and behavior is known as object).
the purpose of function constructor is to create a new function.

Syntax :-
new Function ([arg1 [, arg2 [, ...]]] functionbody)

Parameter
- arguments used by function (eg: size, color, shape)
- fun def on object. (Performed action)

Eg:- (1)

```
<Script>
    var add = new Function ("num1","num2"," return
                num1+num2");
    document.writeln (add(2,5));
</Script>
```

(or)

```
var r = add(2,5);
document.write ("Result =" + r);
```

(2)

```
var power = new Function ("num1", "num2", " return
  Math.pow (num1, num2)");
document.writeln (power(2,3));
```

* Predefined objects: round( ), min(a,b), sin( ), abs( ).
1. Math Eg:- ceil( ), exp( ), floor( ), cos( ), log( ), max(a,b),
2. String
3. Date
4. Array
5. Boolean
6. Number

2) String:- (Att immutable)
two ways to create String objects:-

(i) s = "welcome";

(ii) s = new String ("welcome");

⇒ charAt( ), etc.

Properties of String Object:-

1) length:-
Object.length.

29/8/22

+ 3 ways to construct object:-

1. Object literal => object = { property 1 : value 1 , ... property N : value N }
2. new keyword => var object name = new object ( );
3. object constructor

3) Array :- (mutable) (Heterogeneous)
Array are supported as objects.
Attribute length.
methods include:
Concat, join, pop, push, reverse, sort.

* Advantages:
Array can grow dynamically - just add new elements at the end.
Array elements can be anything (Heterogeneous)
- numbers, string or arrays!

+ Creating an array :
1) new :- var x = new Array (10);
with operator & a size
2) with the new operator and an initial set of element values:
var y = new Array (16, "hi", 22);
3) Assignment of an array literal
var x = [1, 0, 2];

⇒ Eg:-
```
<script>
Names = ["Anurag", "CVSR", 6];
for (var i in Names)
{ document.write (Names[i]);
}
</script>
```

Eg 2:-

```
var a = new Array(4);
for (i=0; i<a.length; i++){
    a[i]=i;
}
for (j in a) {
    document.writeln(j);
}
```

* Array of Array Eg:-

```
var board = [ [1,2,3], [4,5,6], [7,8,9] ];

for (i=0; i<3; i++)
for (j=i; j<3; j++)
```

⇒ concat()
includes()
indexOf()
join()
lastindex of()
length
pop()
push()
reverse()
shift() - removes the first element of an array, and returns the element
slice() - selects a part of array & returns the new array.
sort()
unshift() - adds new elements to the beginning of an array, & returns the new length.

* Strings :- Immutable.

⇒ way to create string :-

1) s = "Anurag"

2) s = new String ("Anurag");

Eg: 1) s.charAt(2)

o/p => u.

⇒ String to Numbers conversion :-

- Global functions.

- parseInt()
  - parses only integers
  - if a string begins with "0x" or "0X", parseInt() interprets it as a hexadecimal number.

Eg:
* parseInt ("3 blind mice"). // => 3.
* parseFloat: ("3.14 meters") // => 3.4
* parseInt ("-12.34") // => -12
* parseInt ("0xFF") // 255
* parseInt ("$72.47"); // => NaN.

⇒ String methods :-

- Var s = "hello, world"

s.charAt(0) => "h"

s.charAt(s.length-1) => "d"

s.substring(1,4)

s.slice(-3)

s.slice(1,4)

s.indexOf("l")

s.lastIndexOf("l")

* Date Object:

4 variant of Date constructor to create date Object.

1) Date()
2) Date (milliseconds)
3) Date (date String)
4) Date (year, month, day, hours, minutes, seconds, milliseconds)

Eg: 1)
```
<script>
var date = new Date();
var day = date.getDate();
var month = date.getMonth()+1;
var year = date.getFullYear();
document.write("<br> Date is :"+day+"/"+month+
"/"+year);
</script>
```

2)
```
<Script>
var date1 = new Date("August 15, 1947 20:22:10
GMT +12:00");
document.writeln("Date value 1: "+date1.getDate()
+"<br>");
document.writeln("Date value 2: "+date1.getDate()
+"<br>");
</script>
```

* Date methods:

getUTCDates(), setDay(), getDate(), getFullYears(), getHours(), getMinutes(), etc.

HTML DOM

=> Document object model.
=> Interface to the JAVASCRIPT

=> change
   access
   remove
} the all HTML attributes

=> access, change, remove the HTML elements
=> JS can add/change/remove CSS styles.

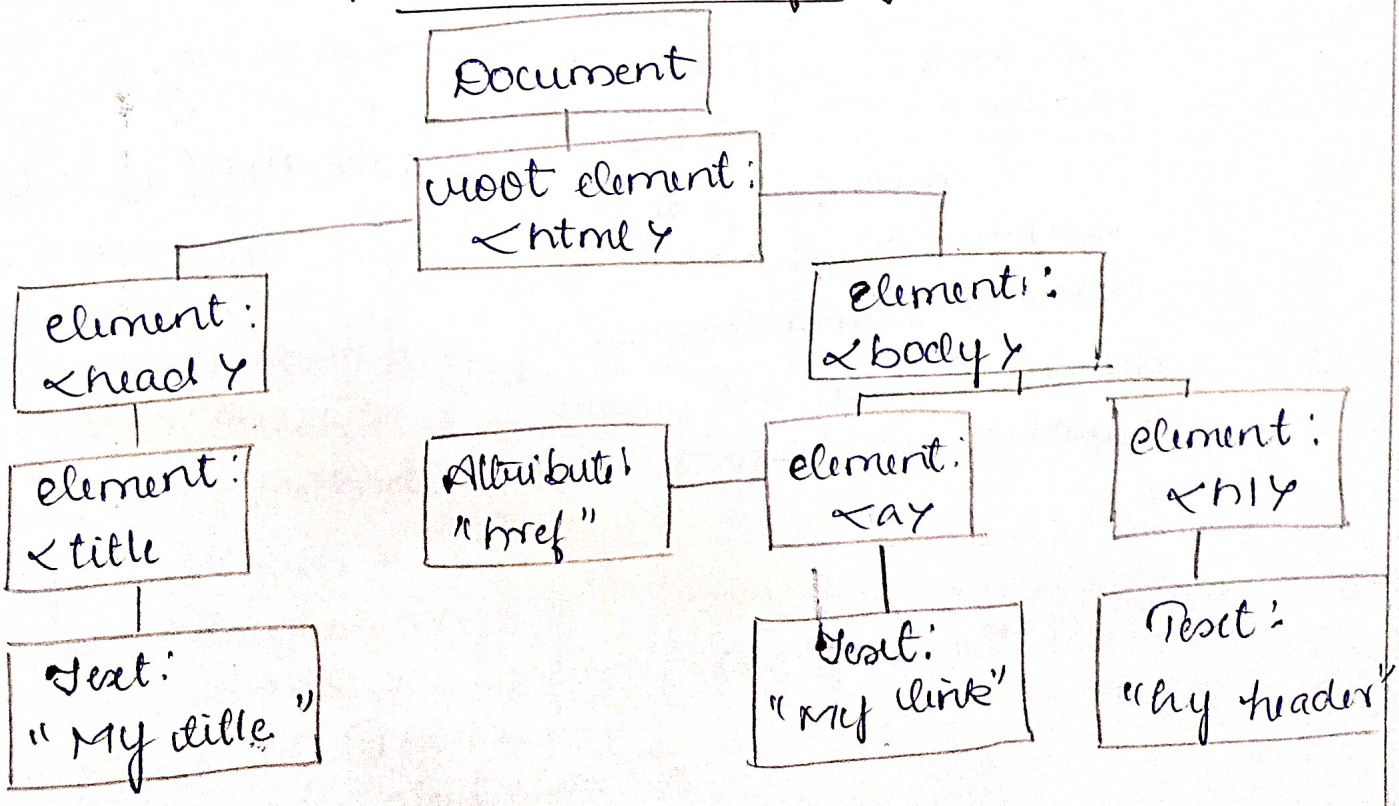
=> properties for HTML elements
=> methods for all HTML elements
=> Events for all HTML elements.

* The HTML DOM is an API (Programming Interface) for JS.

*JS can add/change/remove HTML events.

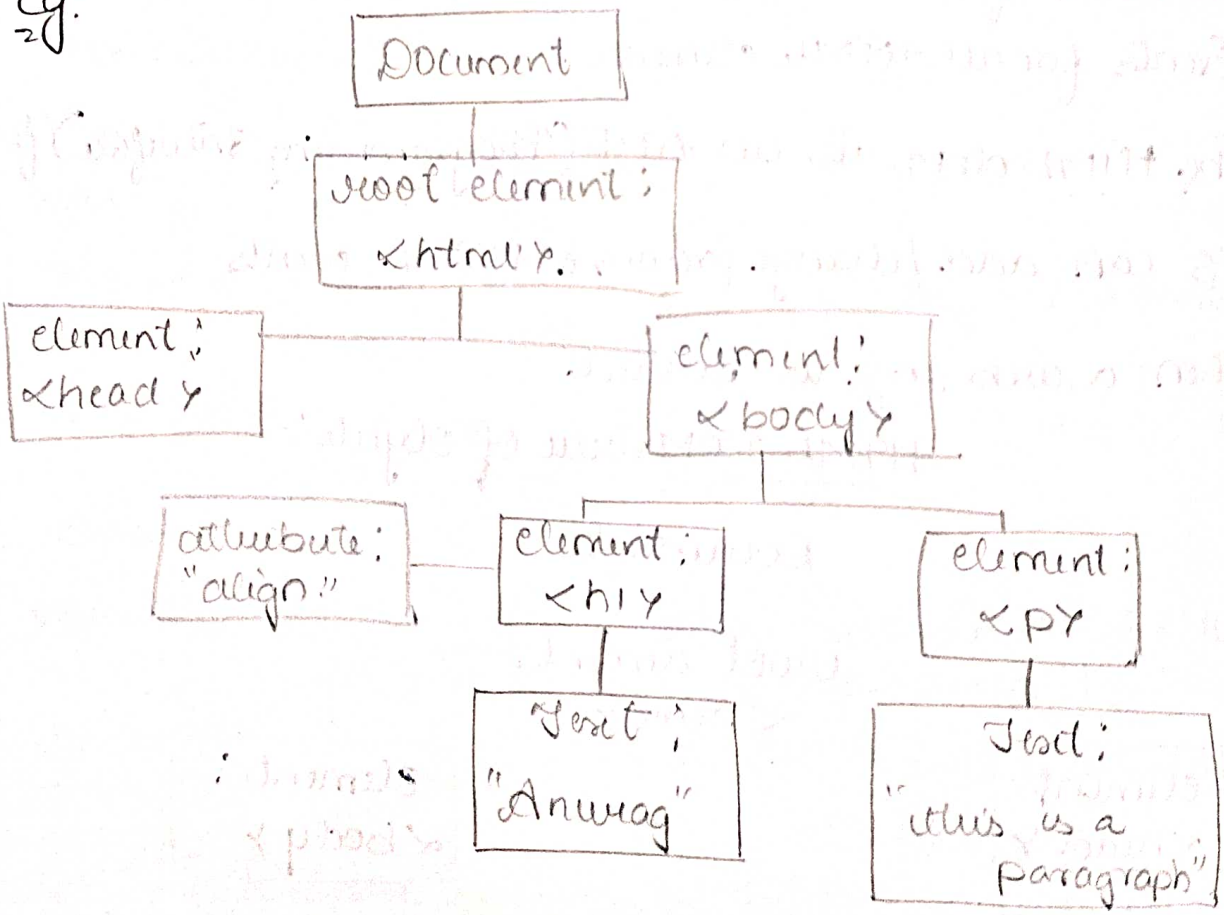
=> when a web page is loaded.

HTML DOM tree of objects:-



```
<html>
 <head>
     <title> My title </title>
 </head>
 <body>
     <a href="#"> My link </a>
   <h1> Myheading </h1>
    </body>
</html>
```

Eg:



2/9/22

* Predefined objects of JS/HTML DOM :-
  - → window
  - → Location
  - → History
  - → Navigation

* window object :-
  - → Supported by all browsers. It represents the browser's window.
  - → Global functions are methods of the window object
    properties variables are properties of window object.

=> Both returns the sizes in pixels :- (window size).

1. window.innerHeight – the inner height

2. window.innerWidth – the inner width.

Eg:
```
<body><p id="demo"></p>
<script>
var w = window.innerWidth;
var h = window.innerHeight;
var x = document.getElementById("demo");
x.innerHTML = "inner window width: " + w + ", height: " +
              h + ".";
</script>
</html>
```

* window methods :-

window.open() - open a new window
window.close() – close the current window
window.moveTo() – move the current window
window.resizeTo() – resize the current window.

confirm() method :- (pop up window is displayed).

Eg :
```
<script>
function msg(){
    var v = confirm("Are you sure?");
    if (v==true){
        alert("OK");
    }
```

```
else {
  alert ("cancel");
}
}
</script>.
<input type="button" value="delete record" onclick="msg()"/>
prompt() method :-
<body>
<p> click the button</p>
<button onclick="myFunction()"> Try it </button>
<p id="demo"></p>
<script>
  function myFunction() {
      var person = prompt("Please enter your name","Anurag");
    if(person != null) {
      document.getElementById("demo").innerHTML=
                                    person;
    }
  }
```

* Location objects :-

The window.location object can be used to get the current page address (URL) and to redirect the browser to a new page.

⇒ properties of location :-

window.location.href → returns href of current page

window.location.hostname → return domain name of web host

window.location.pathname → path & filename of current page.

window.location.protocol → protocol used.

Eg: document.getElementById("demo").innerHTML = "page hostname is" + window.location.hostname;

⇒ Methods of location:

1. assign() — loads a new document
2. reload() — reload the current document
3. replace() — replace the currentdocument with a new one.

Eg:
```
<body>
<button onclick="myFunction">load new document.</button>
<script>
function MyFunction() {
    location.assign("https://www.w3schools.com");
}
</script>
</body>
```

* History Object:

⇒ urls you visited is stored in history object. methods:-

1. history.back() - back in browser
2. history.forward() — forward in browser

Eg:
```
<head>
<script>
function goforward() {
    window.history.forward()
}
</script>
</head>
<body>
input("
</body>
</html>
```

* Navigator object :-

It contains information about the browser.

properties :-

1) appCodeName → returns code name of browser
2) appName → name of browser
3) appVersion → version info of browser
4) cookieEnabled → determines whether cookies are enabled or not.
5) language → language of browser
6) platform
7) product

methods :

1) javaEnabled() :-

browser has enabled or not.

Eg:

```
<Script>
document.writeln("<br/>navigator.appCodeName:"+navigator.
                                    appCodeName);
document.writeln("navigator.appName:"+navigator.
                                    appName);
</Script>
```

5/9/22

* Selecting or Accessing Elements :- (HTML Elements)

1) getElementById() - Selects an element by Id attribute.
2) getElementsByName() - Selects an element by name
3) getElementsByTagName() - Selects an element by tagname.
4) getElementsByClassName() - Select element by one or more class names.

1) getElementById() method:-

Syntax:-

```
document.getElementById(elementID)
```

It is used almost every time you want to manipulate information.

Eg:
```
<body>
<p id="demo"> Amrita University. </p>
<button onclick = "myfunction()">Change</button>
<script>
function myfunction()
{
document.getElementById("demo").innerHTML=
"CSE-C SECTION"
}
</script>
</body>
</html>
```

Eg:
```
<html>
<body>
<p id = "demo"> AU</p>
<p id = "demo"> CUS</p>
<button onclick = "Myfunction()"> change </button>
<script>
function myfunction()
{ document.getElementById("demo").innerHTML=
```

```
"Anurag" ; }
</script>
</body>
</html>
```

8/9/22

* HTML DOM events for JS :-

→ the change in the state of an object is known as an Event.

→ The process of reacting over the events is called Event Handling. Thus, js handles the HTML events via event handlers.

→ Onclick, ondblclick, onfocus, onblur, onsubmit, onmouseover, onmouseout, onmousedown, onmouseup, onload, onunload, onscroll.

* Mouse events :-
Onclick, onmouseover, onmouseout, onmousedown, onmouseup, onmousemove.

* Keyboard events :-
Onkeydown & onkeyup.

* Form events :-
onfocus, onsubmit, onblur, onchange.

difference blw onfocus & onblur

* Window / document events :-
onload, onunload, onresize

* Examples:

=> Mouseover Event

```
<body>
<script>

function mouseoverevent()
{
alert("AU");
}
</script>
<input type="button" onmouse over="mouseoverevent()"
value="Keep cursor over">
</body>
</html>
```

=> Focus Event

```
<body>
<h2> Enter Something </h2>
<input type="text" onfocus="hello(this)">
<script>
function hello(x)
{
x.style.background="yellow";
}
</script>
</body>
```

=> Onblur event

```
<body>
Enter your name: <input type="text" id="fname" onblur=
"myfunction()">
<p> when you leave the input field, a function is triggered with
transforms the input text to
upper case.
</p>

<script>
```

```
function myfunction()
{ var x = document.getElementById("fname");
  x.value = x.value.toUpperCase();
}
</script>
</body>
```

* On keydown:

```
<body>
 <p> using KeyDownEvent </p>
<input type="text" onkeydown="myfunction()">
<script>
 function myfunction( )
  { alert("Hello");
  }
</script> </body>.
```

* Load event:

```
<body onload="myfunction()">
 <h1> Hello World! </h1>
 <script>
  function myfunction()
  {
    alert("Page is loaded");
  }
</script>
</body>
```

* onsubmit :-

```
<body>
<p> when you submit the form, a function is triggered which alerts some text.
</p>
<form action = "index.html" onsubmit = "myfunction()">
enter name : <input type = "text" name = "fname">
<input type = "Submit" value = "Submit">
</form>
<script>
function myfunction()
{
alert ("The form was submitted");
}
</script>
</body>
```

06/9/22

Creating & inserting the data in a document :-

```
x = document.createElement("h1")
y = document.createTextNode("CVSR")
x.appendChild(y).

document.body.appenchild(x)
```

* Creating nodes :-

=> document.createElement()

```
var par = document.createElment('p');
document.createTextNode(text)
```

=> var textNode = document.createTextNode(" ");

⇒ Create Attribute (attributeName).

att = document. Create Attribute ("width");

att . nodeValue = "100";

* Inserting Nodes:-

⇒ appendChild (newNode)

⇒ par.appendChild (text);

⇒ insertBefore (newElement, targetElement)

⇒ insertAfter (newElement, targetElement)

⇒ replaceChild (newChildNode, oldChildNode)

⇒ removeChild (referenceToChildNode)

Eg: x = document.getElementByTagName('P')[0];

document.body.removeChild (x);

* Clone Node :-

duplicate node.