Hyper text Mark Up language(HTML)

Every defined language which uses set of tags through which we could be able to display content on webpage is called as markup languages.

Eg. HTML,XML.

HTML is a predefined markup language comes with a set of predefined tags using which we could able to rendor required content on the webpage.

Note: HTML is the only language through which we can rendor content within the browser.browser only understands html,css,javascript.

Following are set of pre defined html tags using which we could be able to render required content in different ways on the page.

1. Html :Predefined tag to hold the complete content of the page
2. Body: Used to hold the actual body part of page
3. Head: Used to hold extra content of page like title,meta info,external css/js files etc.
4. Title :Used to add title to webpage
5. Div :Block level element to hold block type of content on page
6. Span :Inline element to add inline content of page
7. a: Anchor tag used to create an external reference from current page
8. P: Paragraph tag
9. h1 to h6: predefined heading tag
10. img: used to add a single image resource to page
11. br : break tag used to single line break in content
    - self closing tags
    - doesn’t hold content,so no closing tag eg:</br>
    - bold
    - block level elements
12. UL/OL: Ordered and unordered list tag to hold the group of relative items.

What is tag?

Anu predefined text binded between open and close angular braces is called as tag.

Eg. <div></div>

Html attributes

- Attributes are used to inject or add extra properties(additional information) to any html element.
- Attributes provide additional information about html elements.

Following are set of predefined attributes can be added for html elements.

1.Id: used to add unique identity to elements ,we cannot have same id been assigned to multiple elements

2.Name: used to add name value to any html elements

Eg: <meta name=”description” content=”…”>

here the name attribute specifies a name for the content attribute value

3.Class: using which we could be able to inject any number of css classes to any html elements

```
<style type=”text/css”>

  .className

     {

Properties;
```

```
                        }

                </style>

<html-element  class="“.className”>
```

4.title: used to add title to an element,which will be shown on mouse over of particular html element

5.alt: using which we we can specify an alternate text which we will be shown automically when the resource is not available.

6.href: using which we could be able to specify path of external resource for anchor tag

7.Src: using which we can specify external resource path for img and script tag.

- Syntax for adding attributes to html elements

```
        <tagName attrName1=”value1(user-def)” attrName2=”value2”…>

        ….

        </tagName>
```

Note:

1.for a single tag we can inject any number of attributes,with space as seperator

2.for a single tag,same attribute should not be added for more than once.

3. Same id shouldn’t be assigned to multiple elements

8.Style: Using which we can add css properties to an element.

Different ways of applying css to html elements:

Following are the different ways we can inject required properties to html elements

1.Inline css.(using style attributes)

2.Internal css.(defining a css class under <style>tag.

3.External css.(Creating a external css file which holds all the required css classes)

1. Inline css :

Through style attribute we can inject css properties to a single element.

Note: Inline css is mostly not recommended due to maintenance issue and duplication behavior.

```
Syntax : <tagname style=”cssProperty1:value; cssProperty2:value2;…>

                ……..

                ………

        </tagname>
```

```
Eg. <p style=”color:green; font-size:15px>

            ………….

            …………..
```

```
</p>
```

2.Internal css.(defining a css class under <style>tag.

Css classes (to avoid rewriting code):

- The process of grouping & defining set of css properties as an individual block and assigning user defined name to it,making use of it by injecting its name to a required html element is called a css class
- While defining class names,it is always user defined name.
- Every class name should always start with dot operator while defining it.
- Style is a predefined tag under which we can define any no of css classes.
- Class is an predefined html attribute using which we can inject any no of css classes to any html element.
- Once we define a css class,it can be injected to any no of html elements.
- The same css class can be injected for multiple html elements
- In a single page we can define any no of css classes but it is recommended to create a single style tag throughout the page.(inside head tag)

Syntax:

```
    //define a css class

    .<className> {

    …

    //set of css properties

}

//Injecting a css class to html elements

<div class="className">

…

</div>
```

- Css classes → code reusability & maintainence
- Also for single html any no of classed can be added.

  Class = ".class1 .class2"

Different ways of defining CSS class:

1. Using CSS classname.(already described in the above "css classes" topic)

2.Using elements ID

3.Using tag name of an element.

2.Defining css classes using element ID:

In a case,an html element need to be injected with set of css properties.The set of properties will not be used by any other elements once the element is having an ID property assigned.The elements holds ID attribute,in this case instead of css class using the element ID itself,it is almost like defining a normal css class where the only difference is that css class with an ID starts with hash operator.

Syntax:

```
    #<elementId> {

    …
```

```
        …//set of css properties;

        }

        Eg:

                <style>

                #sample {

                …

                …//set of css properties;

                }

                </style>

                <p id="sample">

                …

                </p>
```

3.Defining css classes using tag name(not recommended)

If multiple same type of element(eg. Multiple ‘p’ tags) need to be rendered with same look end field,then we can define just by using the <tagname> instead of ID or class name.

Note: While defining css with <tagname> it should not be start with any operator or symbol just a <tagname>.

```
    Syntax:

        tagName {

         ……

         ..…//set of css properties;

           }

    Eg.

        <p>

      …….

     …….

     </p>

     <style>

    P{

      Font-size:20px;

      Color:green;

      }

    </style>
```

Frameworks (set of rules and regulations for handling large codeBase)

1.Angular js

- designed for client side
- high performance(only deals with client side)

2.Angular 7:clientSide + server side

- Angular rules and regulations using typescript (superScript of javascript)
- interpreter/compiler(browser cannot process typeScript)→ fed to browser

4.Reactjs:

designed for client side

Adding images within the page

"img" tag is a preDefined tag been supported in html using which we could able to inject a global or a local image resource to the webPage.

It takes a mandatory attribute src using which we can specify the relative or absolute path of an image resource.

Syntax:

```
<img src="relative/absolute path"/>
```

Eg:

```
<img src="http://sample.com/test/abc.jpg"/>

<img src="c:/files/images/data/user.png"/>
```

Note:"Img" is a inline block element ,gets rendered in the same line,but we can control the dimensions using width and height properties.

Interview Questions:

1. Give eg for Self closing tags?
2. Give eg for predefined html attributes?
3. What are the different ways we can apply css to html elements?
4. Why inline and tag based css are not recommended?
5. Difference between a css class and id based questions?
6. Difference between absolute and relative path?

DOM(document object model)

- Which specifies the complete tree structure of the current page.
- Specifies relationship between elements(child,parent,sibling etc)
- It holds list of attributes and its values.
- Every webpage can not have the same DOM structure.
- The dom structure holds list of all the elements,relation between the elements,contents of the elements,attributes it is holding and their corresponding values.

CSS priority order :

Following is the default css priority order any browser maintains while adding css properties to elements.

1. Among all the different ways Inline CSS has the highest priority in the order.
2. CSS being applied through Id of an element takes the second priority in the order.
3. CSS being applied through a CSS class takes third priority in the order.
4. CSS being applied through the <tagname> takes the least priority in the order.

Note: Irrelavant of all above css priority order any css property being added with “!important” takes the heighest presidence among all above.

Eg. Font-size:20px !important;

Debugger tool :

We can easily identify corresponding css and html particular look end field of the webpage.

Using debugger tool we can easily perform CRUD operations on any html element or css properties.

Using this tool we can debug not just the local page but even global pages(amazon,flipkart etc)

There may be different tabs in the debugging tools of diff browsers but all do same job i.e to debug the page.

It is also responsible for making dynamic changes to the webpage but those are temporary changes.

You can debug html,css,js code.

Anchor tag

Is a predefined tag been supported in html using which we could able to create reference from current page to some external web page or any other element within the same page.

It takes a mandatory attribute ‘href’ ,using which we can specify the external webpage link,which need to be linked.

Target is an optional attribute.

Syntax:

```
<a href=”<absolute/relative path of external page>”>
        …link content
</a>
```

Eg:

```
<a href=http://www.google.com>
        Click to visit google page
</a>

Click <a href=http://www.amazon.in>amazon link</a> to visit page
```

Note:

Using anchor tag, we can refer to either local or external webpages.

```
<a href=”filePath/webPageAdresss” target=”_blank”>
```

Adding Image to a page: (img) tag

- It is html predefined tag using which we can inject img resorce to a webpage.
- It takes the mandatory attribute “src” through which we can provide url or path of the img.

Eg. <img src=”http://sample.com/test/abc.png:/> (global image)

Eg. <img src=”c:data/abc.png”/> (local image)

Inline elements and block level elements:

All the elements under html,are been categorized into two types

1.block level elements

2.inline elements

Block level elements:

Any dom element,which comes under the block level category holds following properties.

- While rendering on the page, a block level element automatically comes to a new line ,irrevelant of whether break tag been added or not.
- Any dom element which is following a block level element,also automatically comes to a new line.
- By default , every block level element tries to occupy 100% width of its container.
- Even though it occupies 100%width by default,we can still control the dimensions of it using css width and height properties.
- "Div" tag is the best example for block level elements.

Inline elements:

Any dom element,comes under category of inline elements holds following properties

- inline elements always tries to render within the same line
- it occupies the space(width and height) on the page based on the content it is holding
- we cannot control the dimensions of an inline element with css
- even though ,we try to control the dimensions using css width and height properties,it simply skips and doesn't consider.
- span tag is the best example for an inline element.

Pseudo classes and pseudo elements

Pseudo classes:

Following are set of predefined pseudo classes been supported in css using which we could able to apply css properties to dom elements,not on load of the page but apply the css properties based on the current state of it

1. :hover
2. :active
3. :link
4. :focus
5. :first-child
6. :last-child
7. :nth-child(n)
8. :nth-last-child(n)
9. :checked
10. :empty
    etc

Example

```
        .sample: hover {

        …

          …// set of css properties gets applied to an element not on load of page,but while  a mouse hover happens on the element.

}
```

Pseudo elements:

Through which we can apply css to elements but not to full content only to partial content.

1. ::after
2. ::before
3. ::first-line
4. ::first-letter etc.

Eg. . 1 .abc::first-letter

```
          {
                 ………
                  ………
                }
Eg. 2         #test::after
                   {
                Content:"continue …";
                   }
```

CSS float property

- By default when an element is rendored ,it tries to rendor from top left top corner of page,it follows top to bottom approach if it is block level.left to right approach if it is inline.
- In order to change the default rendering direction of the page or to make multiple block level elements to get rendered within the same line,We make use css property float which take possible values.

css float property

- left
- right

any dom element with float:left property, makes the dom element to get rendered to the complete left direction

float:right will make the dom element to get rendered extreme right direction of the container

multiple continuous dom elements having set the float property to either left or right, all the elements will try to render within the same line irrelevant of whether the elements belongs to inline or block level category.

When a dom element, changes its rendoring direction to either left or right, the dom element which is following the floated element, will also try to follow the previous elements direction.

CSS clear property:

To make the dom elements,to not to follow previous floated element direction ,and to follow its own default direction we make use of the css property.

"clear",which takes following possible values.

- Left
- Right
- Both(recommended)

CSS Margin And Padding Property:

Any time when the dom elements getting rendored on the page it tries to rendor to the very next space of the previous element without maintaining any gap or space.

In order to maintain a gap or space between or with in the dom elements we make use of CSS Margin or Padding properties.

1.Padding:

Using which we could able to maintain space between a border and content of the container.

Eg:

```
  Padding: 5px;//adds padding of 5px in all 4 directions.

 Padding:  5px 10px; //adds padding of 5px on top and bottom

                        10px of padding on left and right of the container.
```

Padding:	10px	5px	6px	11px
	Top	right	bottom	left

2.Margin:

Using which we could able to put a gap or space above the border of a container.

Ex:

Margin-left: 6px;

Margin-right: 5px;

Margin-bottom: 10px

Margin-top: 11px;

Moving any element on the page horizontally we use margin property(not vertically)

Eg. Margin:10px auto;

NOTE:

- ✓ To avoid conflict and redundancy declare a css class having all the properties and link the css file to an html file using <link>.
- ✓ A css file can be used in different html files just u have to give the css file name inside <link>.

Scroll /overflow css property:

We can add scrollbar by using overflow property.

when the size of a container(parent) or box is less than the size of all the elements(child) present inside it we use overflow property to make the scrollable.

Ex: Overflow: auto;(used to add automatic the scroll property when needed into a box)

Css box model:

Any time the new element getting rendered on the page it gets rendered after the existing element of the page.

The box model specifies(calculates) the actual dimension or total space the existing dom element has occupied on the page, so that the other dom elements gets rendored accordingly.

Using box model when browser calculates the actual dimension of an element it considers following properties of the elements.

1. Total padding space been taken in all directions.
2. Total margin being occupied,
3. Total border space been occupied.
4. Its own actual dimensions (height and width properties).

CSS positions:

- In order to move any dom element to any required position we make use of css padding or margin properties.
- If we make use of margin or padding properties to move the dom elements, we are not actually moving the dom elements instead we are increasing the dimensions of the dom elements.
- Following are the css properties we can actually move the dom elements to any required position without increasing the dimensions.
    - Top
    - Left
    - Right
    - bottom
- The above css properties can’t be applied on every dom element but can be applied to the elements whose position is non static.

CSS position property:

- This is the property through which we can control any dom elements position with in the page.

- Following the possible values of this property:
  - Static
  - Relative
  - Absolute
  - Fixed
  - Sticky

1.Element with position Static:

By default every dom element holds its position as static which indicates the dom element can not be move to any required position (even though we apply css properties like top, left, right or bottom, it completely skips it).

Syntax: position: static;

As it does not move to any other position it occupies only its previous XY axis and does not move to Z axis.so we can not control the Z index of static element.

2. Element with position Relative:

Any dom element holding the css position value as relative holds following properties.

- It is capable of moving to any required position within the page (it considers top, left, right or bottom properties)
- While moving to a new position it never loses the default space been occupied on load of the page.
- While moving to new position it always moves relative to its actual default relative position.

3. Element with position Absolute:

Any dom element holding the position absolute holds the following properties;

- It is capable of moving to any required position within the page.
- It never holds its actual default space with in the page.
- While moving to a new position it always moves relative to its parent's position.
- While depending on a parent position it only depends on the parent whose position value is non static.
- If its immediate parent's position is static it keeps traversing to its grandparents until it finds an element non-static.
- If none of its parents holds the position value as nonstatic it tries to depend on the position of body tag.

Note:

By default every static element on a page gets rendered with in x-y axis.
Elements with position non static automatically jumps from default x-y axis to z-axis.

4.Elements with position fixed

Any dom element having the position as fixed , is almost like absolute , doesn't hold its position while moving to a new space,it always depends on the body tag while moving to a new position,once the position of a fixed element is been assigned it doesn't even move its position even while scrolling.(eg:header part in any webpage)

Syntax: Position:fixed;

5.Element with position sticky

Any dom element having position as sticky is almost like any an element with position relative holds following properties

1.while moving to a new position , it doesn't lose its actual space on the container.

2.while scrolling if the element is about to come out of its viewport it automatically turns to fixed element and doesn't gets scrolled.

Css z index property

When the dom elements position is being changed from static to non static , they automatically jumps from default xy axis to z axis.

When the elements gets rendored on z axis,there is a chance of gets overRide each other

When elements gets override each other,we can control the order of rendoring using css property z-index.

Element with higher Zindex value always rendor on the top.

z-index is the css property which takes an integer value between 1 to any bigger integer(9999)

The z index property can only be assigned for a non static elements.

Any non static dom element having higher z index value rendors on the top

Eg:pop Up

Q&A

What is the use of css position property ?

What are the possible values a position attribute takes ?

Difference between relative and absolute elements ?

Difference between sticky and fixed elements ?

How do we control the rendering order of non static elements ?

What is css z index property ?

Node JS overview

Node JS is a server which uses JS code.

Why we need server?

- To get the data from database.
- To hold resouces.

Examples of server: Node JS(uses JS), apache(uses java), jboss(uses java), .net, django(uses python)

We should not store our files under drives as it can not be accessed or viewed by other system.so we must save it under our user defined server so that we can share it to other.we can share it by giving the URL.

Steps to install NodeJS:

- Download and install nodejs setup file.
  https://nodejs.org/en/
- To check the successful installation of nodejs open command prompt and give the dos command
  Node –v
  It will tell you the node version.
- Install express and express generator module by giving the following commands
  -npm install express –g
  -npm install express –generator –g
- Create a server using express by passing server name(user defined) to express command.
  Eg. express web_app(user defined name).
- After this, a new folder will be automatically created with the given name with list of predefined files and folders.
- Under the command prompt go into the folder until the server comes and give the below command to install dependency node module.
  -npm install
- Now under the folder open app.js file and give the following instructions

```
app.listen(8086,function(){
        consol.log("server is listening at8086")

         });
```

- Go into the command orompt and give the command to start the server.
  -npm start
- To check the succesfull start of server pick the url in browser which shows "welcome express message"
  Localhost:8086

Note:

- Once you install the server you can create any nos. of projects inside it.
- delete the unneccesary predefined folders inside public folder.
- Now place your files under "public".
- To change localhost to hostname enter a command
  -hostname
- Enter "cls" command to clear the prompt.

Opacity

A css property through which we could able to control the transparency levels of any dom element which takes a value between 0 to 1 where 0 indicates complete transparency and the element will not be visible.

1 indicates no transparency.every dom element by default holds the opacity property as 1

Eg:

Opacity: 0.2;

Html tables

Following are the predefined tags been supported in html using which could able to render the content in the form of row wise and column wise.

1. <table> – parent tag of table structure,holds the complete table
2. <tr> – indicates table row,used to hold the complete row of a table
3. <td> – table data
4. <thead> – block to hold complete header block
5. <tbody> – block to hold complete table body structure
6. <tfoot> – holds table footer
7. <th> – table header cell

Following are the attributes can be added to a table tag

Cellpadding & cellspacing – the two attributes using which we could able to maintain equal space between and within the table cells.

- Colspan is used for merging column wise two or more elements.
- Rowspan is used for merging row wise two or more element.
  Eg. <td rowspan="2">hyd</td>

Structure of table tag

```
<table>

   <thead>

      <tr>

           <th>…</th>

            <th>…</th>
```

```
        </tr>
    </thead>
<tbody>
     <tr>
           <td>…</td>
            <td>…..</td>
       </tr>
</tbody>
<tfoot>
    <tr>
         <td>…</td>
   </tr>
</tfoot>
</table>
```

Note: It is recommended to not to use table tag to rendor data in row wise and column wise.we can use either div tag or li tag.

In order to rendor content on the page browser takes more time as it needs to maintain structure of thetable in turn it decrease the performance of page.

- **Background-image :** it is a css property through which we can set img as a background to a container.
  Syntax: background-image:url(path)
- **Background-repeat:** is a css property through which we can control the repeativeness of the background img.
  Eg.background-repeat:no-repeat;
  background-repeat:no-repeat; // by default
  background-repeat:repeat-X; //repeat on x-axis
  background-repeat:repeat-Y; //repeat on Y axix
- **Background-position :** it control the position.
  Eg. background-position:10px 20px;

Image spriting:

- The concept of merging multiple static images to form a single image using css property background-position we show only the required image is called as image spriting.
- Using this we can increase the performance of the page by not heating the server for multiple time instead start the server only once to get the sprite image.
- This concept can be used for only static images not for dynamic images(eg.product details) bcoz it changes regularly.

CSS display property

Display is a predefined property through which we can change the default display property of an element.

Following are the possible values a display property takes

1. None
2. Block
3. Inline

4. Inline-block
5. Flex(used in css3)

1. **display with none property:**

   through which we can make a dom element not to be display on page but it exists on dom structure.it does not even hold its space on page.
   What is the need?
   Whenever we want to display a error message it will be shown bt until that it will be under display none property.

   Syntax: display:none;

2. **Elements with display block:**

   using which we could able to force any dom element(even inline element) to get rendered in the form of a block level element i.e we will be able to change the height and width of the inline element by making it block level.

   Syntax: display:block;

3. **Element with display inline:**

   using which we could able to force any dom element(even block element) to get rendered in the form of a inline level element.(in the same line)

   syntax: display:inline;

   note: further when we will create our own customized tags to make them either block level or inline level elements display:block and display:inline will be used.

4. **Element with display inline block:**

   any element having css propery as display inline block makes the element to get rendered as like an inline element but capable of controlling its dimensions as like a block level element.

   Syntax: display:inline-block;

5. **Element with flex:**

   Introduced in css3 through which we can add more flexibility to elements.

Difference between display none and visibility hidden property

Display none and visibilty hidden are the css properties through which we can make the dom elements to be not visible on the page even though they are part of dom structure .

Display none makes the dom element to be not visible on the page,it doesn’t even occupy the space of the container ,where as visibility hidden makes the dom element to be not visible at the same time it still occupies the actual space of the element.

Questions:

1.In order to render the data in the form of row wise and column wise which tags you prefer?

2.How does the table tag decreases the performance of a page

3.What are the possible values css display property takes

4.How do we force an inline element to render like a block level and vice versa

Form elements

Html input elements:

Following are set of predefined html tags been supported using which we could able to read different types of data from the user.

Eg:

1. <input type="text"> used to read text type of content
2. <input type="password"> used to read confidential info
3. <input type="number" min=".." max=".."> used to read number from a given value to max given value.
4. <input type="phone"> reads a phone no.
5. <input type="file"> reads a file and send it to the server but for that we need to declare an attribute inside the form i.e:

6. <form action="return.html" method="GET/POST" enctype="multipart/from-data">

7. <input type="hidden">
8. <input type="url">takes an url.
9. <input type="color"> takes color.
10. <select>
    <option>…</option>
    <option>…</option>
11. </select> :- creates a dropdown element with multiple values, from which user can select one.

12. <input type="radio"> creates a radio button
13. <input type="checkbox">creates a check box
14. <input type="button" value="lable"> creates a button element, which we can take user action as input.
15. <input type="submit" value="….">Creates a button, will be used in form tag, to send data to specified server.
16. <input type="reset" value="…."> resets the values.
17. <textarea></textarea> creates a multiline text box, to give lengthy text content

Note:

'Placeholder' attribute is used instead of 'value' attribute to give instructions inside the input area.

Required attribute forces the user to insert the value.

Eg: <label>Username</label>

<input type="text" placeholder="username" required>

Etc.

Sending client side data to server using Form tag:

Form is a predefined html tag capable of automatically reading the data from the input elements within it and send it to the specified server.it takes the following two mandatory attributes.

- Action
- method

Action attribute: which takes a server path or url to which data need to be sent.

Method attribute: It specifies what method need to be followed while transmitting data to the server.

following are the possible values method attribute takes

1.GET

2.POST

'GET' type of communication:

It is a non-secured type of communication in which data will be sent to server in the form of query parameters (by appending user data to the url itself).

Eg:

http://abc.com/data/sample?name=raj&age30

http://sample.com/data?age=22&course=mca&pwd=india

Name attribute should be mandatory in GET type of communication in order to make our values been entered to be seen in the query parameter bcoz it will create key and value.

Eg. <input type="text" name="uname">user name

Post type of communication:

it is a secure of communicating to the server in which the data been sent to the server will not be visible in the url but it will be sent to the server by adding the data into the request header itself

post is the secure channel has to be used while communicating with confidential information.

Declaration: <form action="url" method="GET/POST">

1. What are the input elements you have experience with?
2. What are the different ways we can send data to server?
3. What are the types of communications we can follow while communicating to the server?
4. Difference between get and post communication?
5. What do you mean by query parameters?

6.What are the limitations of form tag?

If the data has not been entered then also it can send to server so many data has been missed to be entered such as password,age,etc.

Javascript

It is a client side scripting language capable of getting executed within the browser itself using which we can make the applications to behave dynamic and render according to user action

Using javascript we can add arithmatic or logic ability to the web pages.

Every browser by default comes with individual js engine using which it compiles and executes the javascript instructions at the client side itself without any dependency of any servers.

It is a superheroic programming language capable of implementing and performing any type of CURD operations on any html elements or its corresponding css properties

Its comes with set of pre defined objects and reusable methods using which we can add event handling to the dom elements so that application behaves according to the user action.

It supports some predefined objects using which AJAX communication can be established.

It supports some predefined data structures(JSON,arrays,maps,sets etc) thrugh which data within the application can be operated.

It supports predefined control structure through which logical ability to the application can be added.

Adding or injecting javascript code to the webpage

<Script>is a predefined tag been supported in html using which we can inject inline or external javascripts file to the page.

In an single application we can include any no of external javascript files by using multiple script tags.

Inline js:

```
<script type="text/javascript">

…

…// inline javascript code

</script>
```

External js:

<script type=”text/javascript” src=”external js file path” ></script>

Data types

In javascript it is not mandatory to specify datatype while declaring a variable before making using of it.

‘var’ is a predefined key word been supported in javascript using which we can declare variables in javascript

A variable been declared in js is capable of holding following types of data

1. Number(includes both decimals and integers)
2. String (~~char)
3. Boolean
4. Undefined(value not assigned)
5. Null
6. Object
7. Functions(is also a datatype)

In javascript it is not needed to define the datatype of a variable while declaring it,which just make use of var keyword while declaring it.

Javascript has the intelligence of automatically identifing the datatype of variable data based on type of data it is holding

Eg.

```
<script type=”text/javascript”>

        Var firstValue,secondValue=9,thirdValue,result;

        firstValue = 4;

        thirdValue = 5;

        result ;

        result= firstValue+ thirdValue;

        alert(“”the sum is” + result);

</script>
```

“Typeof” method

A typeof method in javascript takes a variable as a input and returns type of data it is holding

Syntax:

Typeof(<variable name>);

Eg

Var a = 90;

Typeof(a) – returns number

Write a program to work with employee details like name ,department,gender, basic salary And calculate the total salary of the employee where

Tsal = pf + hra + basic salary

Note: pf is 14% basic salary

Hr is 25% of basic salary

```
Var name = “Mr.X”,department = “testing”,gender=”male”,bSalary = 1000,totalSalary;

pf =  (14 * bSalary) / 100;

hra = (25 * bSalary) /100;

totalSalary = pf + hra + basicSalary;

console.log(totalSalary);
```

Typecasting

The process of converting datatype of a variable to another datatype is called typecasting

Following are the predefined methods through which we can convert datatypes one to other:

1. parseInt(value) : returns number
2. parseFloat(value) : returns decimal number

Dynamic typecasting(default type casting)

Javascript internally supports dynamic typecasting feature which means it automatically changes datatype of a variable to another based on the value it is holding

In c language value of a variable can be updated but not its datatype.

Eg.

```
Var a=10;

Typeof(a); // returns number

 a = “Raj”;

typeof(a); // automatically changes the datatype of a from number to string and returns string.
```

Prompt : method to read data from user.

It is a predefined method supported in JS using which we could able to read value from user and assign it to variable.

Eg. var a=prompt(“enter value of a”);

variable hoisting

the concept of identifying all the variable declaration, moving them to the starting of the application is called variable hoisting.

Note:while moving the declarations to the starting of the application it only moves the variable declarations,but not the initialized values.

In most of the languages it is mandatory to declare all the variable names only at the starting of the application.whereas in javascript It provides a privelege to declare variables where ever we wanted within the application

In javascript it has a predefined feature of automatically finding all the variable declarations within the application and moves to the starting of the application which is called as variable hoisting

Difference between “=” ,”= =” , “= = =”

“=”

It is an assignment operator using which we can assign values to variables

Eg. var a=10;

“= =”

It is a comparison operator using which we can compare the left side with the value of right side.it returns true or false.It only checks for value equality but does not look for datatype equality.

Eg. a= 45;

b=45;

A= =b ;// returns true

“= = =”

It is a comparison operator used to compare values as like “= = “ operator where the only difference is while comparing the values it not just check for value equality but it also checks for datatype equality

Eg. var a=45;

Var b=”45”

a= = =b; // false

Control structures

Js supports predefined control structures using which we could able to control the normal execution flow in the way we want.

Following are three control structures being supported in JS:

1. Conditional control structures
   - If
   - If else
   - Nested if
   - Nested if else
   - Else if

2. Looping control structures
   - For loop
   - While
   - Do while
   - For of
   - For in
   - For each

3. Case control structures
   - Switch case

1.Conditional control structure

It is used to execute the set of instructions only if the given condion is true.

a. if :

```
syntax:  if(condition)
             {
             }
```

b. if else :

```
syntax: if (condition)
           {
           }else
            {
             }
```

c. nested if :

```
syntax:  if(condition)
   {
           …..// if statemrnt
      If()
              {
                      ……//nested if statement;
                }
            }
     Else{
            …..//else block;
            }
```

d. nested if else:

```
syntax:  if()
      {
            If()
            {
             }else if{
               }
            Else{
                    }
                }else{
```

}

2.Looping control structures

It is used to repeat the execution of code more than once.

a. for loop:

```
syntax: for(initialization;condition;increment/decrement)

                   {

                          …….//statements;

                    }
```

b. while:

syntax:

```
while(condition)
        {
        …..
        …….//set of
instruction;
}
```

c. do while :

```
syntax:  do
                {
                  …..
                }

While(condition);
```

3. case control structrure

```
Syntax:
switch(variable)

 {
 Case "value1":
   …….
.//statement;
Break;

Case "value2":
   …….
.//statement;
Break;
.
.
.
.default:
…….
……..
}
```

Javascript functions

The concept of defining set of instructions as an individual module assigning a user defined name to it, invoking the set of instructions wherever we want it in our application by calling the function name. is called functions.

1. function is a predefined keyword supported in javascript using which we can create a function in javascript
2. using functions we can avoid redundancy or duplication of code or achieve reusability of code.
3. once we define a function it can be invoked any number of times within the same application
3. in a single application we can define any number of functions.
4. javascript does not support defining multiple functions with the same name within the same application(function overloading)

Syntax:

```
      Function <user defined
function name>(optional
parameters)
{
…..
…..
}
```

Eg.
```
function display()
{
Console.log("test");
}
```

Note:
as function is a datatype supported in javascript functions can be assigned to variables.

Eg.
```
var display= function()
{
….
….
}
```

Functions can be called anywhere within the applications,it can be called within the same function definition,the concept of calling the function within the same function is called recursive function.

Scope of variables

It specifies the accessibility level of variables it could be either local scope or global scope.

1. global scope variables(public scope variables)

The varibles which are declared out of any module or block are called global variables.
These variables can be accessible or can be modified anywhere withn the application.

2. local variables:

The variables which are declared within the module or function are called local variables.
These variables can not be accessible outside of the module.

Eg.
```
<script>
    Var age=67; //global
variables
    ……
…….
Function sampleData()
{
……
….
Var a=45; //local variables
Var b=34;
……
……
}
</script>
```